# LOAN DOCUMENT

PHOTOGRAPH THIS SHEET

DTIC ACCESSION NUMBER

LEVEL

INVENTORY

NADC-76011-20

DOCUMENT IDENTIFICATION

DISTRIBUTION STATEMENT

ACCESSION FOR
NTIS GRA&I
DTIC TRAC
UNANNOUNCED
JUSTIFICATION

BY
DISTRIBUTION/
AVAILABILITY CODES

| DISTRIBUTION | AVAILABILITY AND/OR SPECIAL | |
|---|---|---|
| A-1 | | |

DISTRIBUTION STAMP

DATE ACCESSIONED

DATE RETURNED

19981223 042

DATE RECEIVED IN DTIC

REGISTERED OR CERTIFIED NUMBER

HANDLE WITH CARE

PHOTOGRAPH THIS SHEET AND RETURN TO DTIC-FDAC

DTIC FORM JUN 90 70A

DOCUMENT PROCESSING SHEET

PREVIOUS EDITIONS MAY BE USED UNTIL STOCK IS EXHAUSTED.

LOAN DOCUMENT

NADC- ...

ON-LINE ACOUSTIC ANALYSIS SYSTEM
(OLAAS)

PROCESSING & DISPLAY SYSTEM

N62269-77-C-0533

NADC

Tech. Info.

NO DISTRIBUTION
STATEMENT

8000282

BUNKER
RAMO

ELECTRONIC SYSTEMS DIVISION

OLAAS PROCESSING & DISPLAY SYSTEM

SOFTWARE DESIGN NOTEBOOK

## TABLE OF CONTENTS


SECTION

FOREWORD

1.0 INTRODUCTION

- 1.1 OLAAS Software System Configuration
- 1.2 Software Features
- 1.3 Communication Interface

2.0 OPERATING INSTRUCTION

- 2.1 Introduction
- 2.2 OLAAS System Hardware Configuration
- 2.3 OLAAS System Operation
- 2.4 Bootstrap Loading Procedures
- 2.5 Coreload Generation Procedure
- 2.6 Fourier Analyzer Program (Keyboard Stack)
- 2.7 Disc Copy Procedure (MADIU Program)

3.0 OLAAS DISC SYSTEM

- 3.1 Introduction
- 3.2 OLAAS Fourier Disc Layout
- 3.3 OLAAS Fourier Disc Generation

4.0 REFERENCES

5.0 OLAAS SOFTWARE DOCUMENTATION

- 5.1 Software Overview
- 5.2 OLAAS Processing & Display System
- 5.3 OLAAS Processing & Display System Subroutines
- 5.4 OLAAS User Programs & Subroutines
- 5.5 OLAAS System/Coreload Transfer Files
- 5.6 OLAAS System/Coreload Command Files
- 5.7 OLAAS System/Coreload PCS Input Files

APPENDIX

A OLAAS Acceptance Test Plan

B FMTXX1 Program Listing

C Disc Directory For The RTE System Disc (Lower Disc)

OLAAS PROCESSING AND DISPLAY SYSTEM

SOFTWARE DESIGN NOTEBOOK

## FOREWORD

Bunker Ramo Corporation, Electronic Systems Division has been engaged in the development of the ON-LINE ACOUSTIC ANALYSIS SYSTEM (OLAAS). The OLAAS System is a computerized data analysis system for the ERAPS Sonobuoy Advance Development Model.

The OLAAS PROCESSING AND DISPLAY SYSTEM allows an operator to collect and analyze ERAPS data subject to user-selected parameters. The user/operator may select the number of beams analyzed, look directions, range and doppler window. The type of display plotting is also under the user/operator control.

This document is submitted in accordance with, and as a completion of the following data items:

Contract N62269-75-C-0418, Item 0005

- A004, Documentation of Computer Programs
- A005, Operation and Maintenance Handbooks
- A006, Detection Algorithm Description
- A007, System Performance Specification

Contract N62269-77-C-0533

- A002, Program Description Document

OLAAS PROCESSING AND DISPLAY SYSTEM

# SOFTWARE DESIGN NOTEBOOK

## 1.0 INTRODUCTION

The OLAAS PROCESSING AND DISPLAY SYSTEM, Software Design Notebook, describes the documentation for On-Line Acoustic Analysis System (OLAAS System). This document consists of the programs and subroutines that make up the system, operating instructions for system operation and generation, and additional details pertaining to the overall system.

## 1.1 OLAAS SOFTWARE SYSTEM CONFIGURATION

The OLAAS Software System uses the Hewlett Packard 5451B FOURIER ANALYZER System hardware, including a HP7900 Disc Memory. Because of the size of the OLAAS System programs, and a need to maintain an initialized version of the analysis and data collection programs, it was necessary to segment the programs into separate coreloads. The HP2100S Computer has 32K of core memory, and the total system can accommodate 8 Fourier coreloads, using the HP7800A Disc Memory. The coreloads are organized as shown:

| Coreload | Function |
|---|---|
| 0 | OLAAS Executive Program |
| 1 | OLAAS Analysis and System Executive Subroutine |
| 2 | OLAAS Raw Data Collection Subroutine |
| 3 | Copy of coreload 1 (used for processing) |
| 4 | Copy of coreload 2 (used for multi-ping recording) |
| 5 | OLAAS User Programs Analysis |
| 6 | Fourier Analyzer Graphics |
| 7 | OLAAS Plot Display Subroutines |

The key to the system is the transfering of control from one coreload to another (core swapping or overlay).

## 1.2 SOFTWARE FEATURES

The major feature of the OLAAS System is that it is completely operator oriented, i.e.; the operator is prompted on the display screen with messages requesting inputs for each action to take place or parameter to be used. The system is verstile in that the operator may select any option available as desired. The one restriction in the system is that raw data has been collected and stored on the disc. Additional features are that the Hewlett Packard software is used to generate and change programs and create the OLAAS Fourier System Disc.

## 1.3 COMMUNICATION INTERFACE

The operator's communication with the OLAAS System is in either of two (2) methods. (1) Input through the CRT TERMINAL Keyboard is reply to prompt messages displayed on the screen. (2) operate the PH2100S Computer Operator Panel or the 5451A Fourier Analyzer Keyboard. Using the procedures in Section 2.0, Operating Instructions, will describe the use of these methods.

# 2.0 OPERATING INSTRUCTIONS

## 2.1 INTRODUCTION

Operation of the OLAAS System is through the CRT Keyboard and the control switches on the HP-2100 Computer front panel and the HP-5451B Fourier Analyzer front panel.

2.1.1 Conventions. Certain conventions apply to all keyboard entries, as follows:

(1) For numeric inputs, a permissible range is usually printed within parentheses: (Ø-1ØØØØ). Any value outside this range will result in a repeated request to enter the number. The same will occur if an illegal character is entered.

(2) For numeric inputs, the current value is also printed within parentheses. This value will remain unchanged if RETURN is pressed (no characters entered).

(3) All numeric inputs are integers (i.e., no fractional part) unless a decimal point appears in the range of current value printouts. For integers, the decimal point must be omitted; for other numbers, the decimal point may be omitted if there is no fractional part (tenths, hundredths, etc.).

(4) For all numeric inputs, leading zeros, trailing zeros, and the sign, if positive, may be omitted.

(5) All inputs are completed by pressing RETURN. The use of the RETURN key is assumed and will not be repeated in the operating instructions.

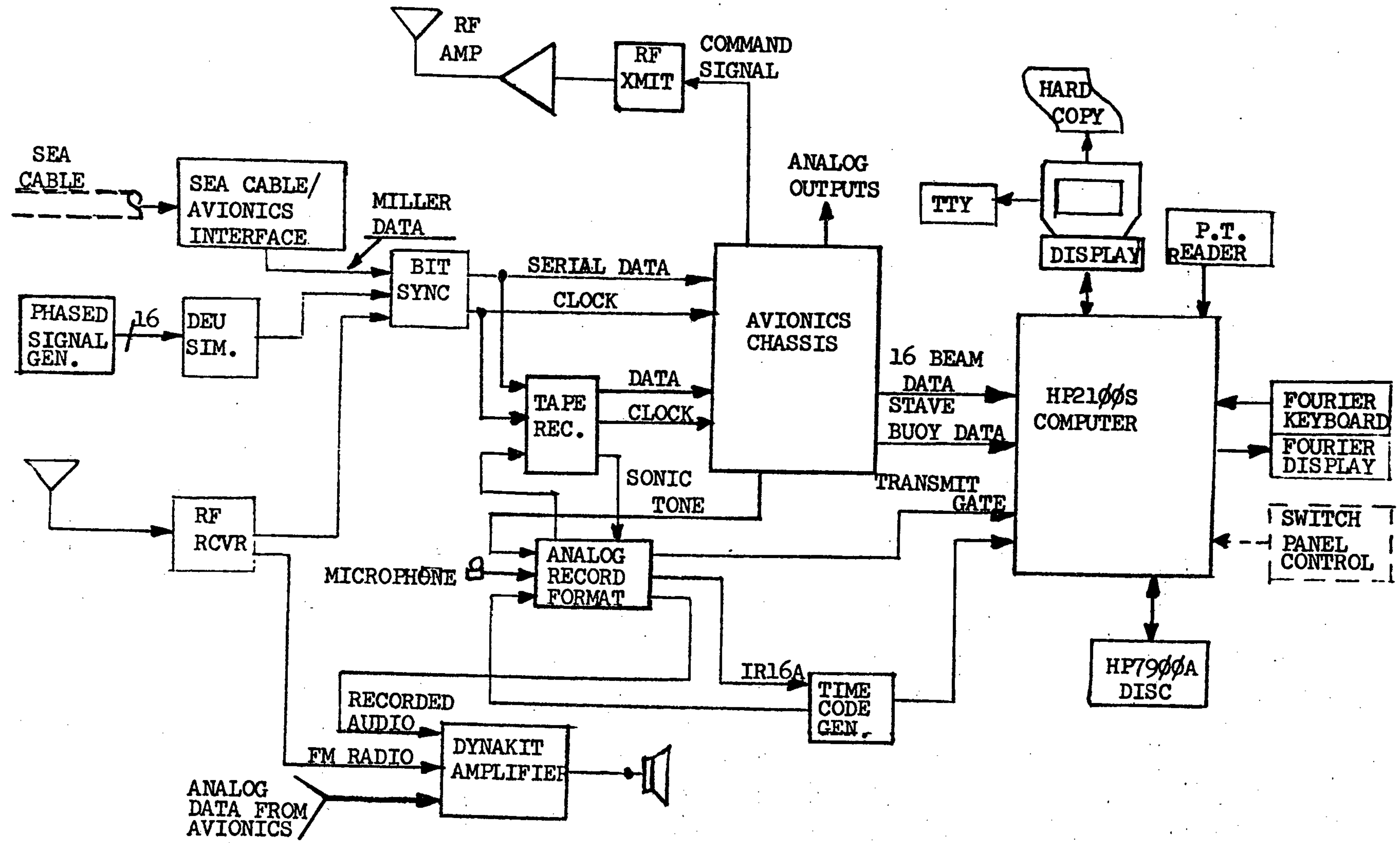
(6) An error recognized before pressing RETURN may be corrected by pressing RUBOUT, then making the proper entry.

(7) Entering 12345 in place of most numeric inputs will result in control being returned to the Executive.

(8) Entering 11111 in place of most numeric inputs when doing DATA RECORDING & ANALYSIS will result in control being returned to the start of processing and will save analyzed data.

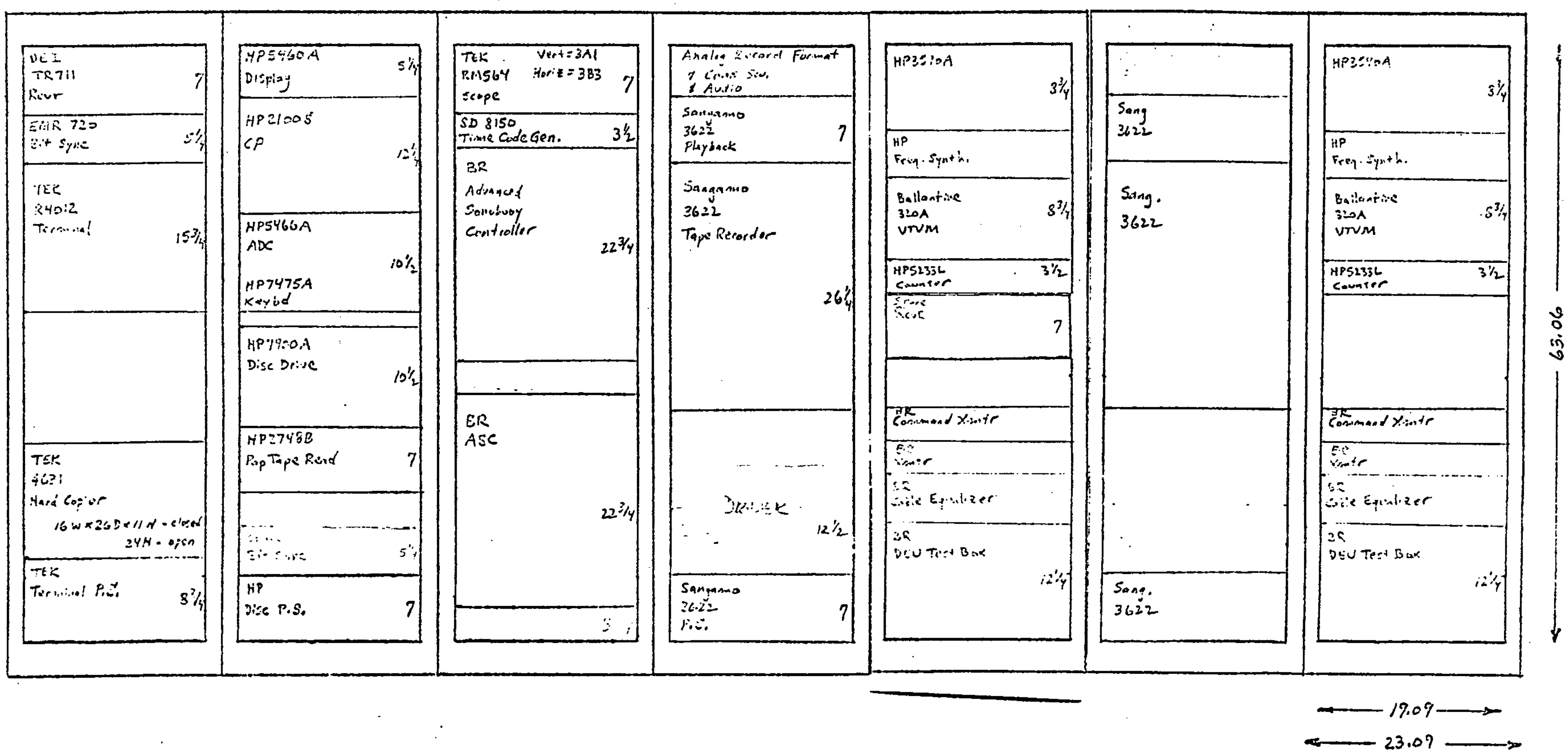
## 2.2 OLAAS SYSTEM HARDWARE CONFIGURATION

The OLAAS Hardware Configuration is shown in the block diagram, Figure 2.2-1, and the ERAPS Van layout is shown in the block diagram, Figure 2.2-2. Turn-On and operation of the OLAAS System is described elsewhere.




OLAAS VAN SIGNAL FLOW

FIGURE: 2.2-1




TTY = 21½ W × 19 D × 10 H

Ea. ¼" square = 3½ in.

Figure 2.2-2: ERAPS Van Hardware Configuration

## 2.3. OLAAS SYSTEM OPERATION PROCEDURE

2.3.1. Ensure that power is on at all applicable units in the van.

2.3.2. Ensure that the OLAAS SYSTEM Fourier Disc Cartridge is installed and that the disc is ready (DISC READY light is illuminated).

2.3.3. Execute OPERATION PROCEDURE: 2.4.1 - OLAAS System Loading Procedure.

2.3.4. The OLAAS EXECUTIVE program was entered by "bootstrap" loading the system and the following message was displayed:

```
***** OLAAS PROCESSING & DISPLAY SYSTEM *****
              EXECUTIVE PROGRAM

         ** ENTER DESIRED OPERATION **

         D = DATA RECORDING & ANALYSIS
         F = USER PROGRAM ANALYZER
         G = FOURIER ANALYZER GRAPHICS
```

2.3.5. Enter the indicated letter for the desired operation and execute the procedure specified in the following paragraphs:

| | |
|---|---|
| D = DATA RECORDING AND ANALYSIS | = 2.3.6 |
| F = USER PROGRAM ANALYSIS | = 2.3.7 |
| G = FOURIER ANALYZER GRAPHICS | = 2.3.8 |

2.3.6. Data Recording and Analysis Operation.

2.3.6.1. Enter the letter "D" for the DATA RECORDING ANALYSIS operation and the following message will be displayed:

```
***** OLAAS PROCESSING & DISPLAY SYSTEM *****
              EXECUTIVE PROGRAM

         ** ENTER DESIRED OPERATION **

         D = DATA RECORDING & ANALYSIS
         F = USER PROGRAM ANALYZER
         G = FOURIER ANALYZER GRAPHICS
D

    ***** DATA RECORDING & ANALYSIS *****

            SYSTEM INITIALIZATION

OPERATOR NOTE:  SET SENSE SWITCH 15 ON TO CHANGE THE BUOY
                FREQUENCY, PULSE LENGTH, & CW NOTCH FILTER.

ENTER DATE
(                  )
```

2.3.6.2. If the operator sets Sense Switch 15 on the indicated question will appear in their proper position. These procedures assume the switch is off.

2.3.6.3. Enter the desired date and the following message will be displayed.

```
***** OLAAS DATA RECORDING & ANALYSIS *****
           25 JANUARY 1978

      ** ENTER DESIRED FUNCTION **
1 = ON-LINE DATA RECORDING & ANALYSIS
2 = OFF-LINE ANALYSIS & PLOT DISPLAY
3 = DATE ENTRY OR CHANGE                             (    2)
```

2.3.6.4. Enter the indicated value for the desired function and execute the procedure specified by the following paragraphs:

```
1 = ON-LINE DATA RECORDING AND ANALYSIS    = 2.3.6.4.1
2 = OFF-LINE ANALYSIS AND DISPLAY          = 2.3.6.4.2
3 = DATE ENTRY OR CHANGE                   = 2.3.6.4.3
```

2.3.6.4.1 On-Line Data Recording and Analysis Function.

2.3.6.4.1.1 Enter the value "1" for the ON-LINE DATA RECORDING AND ANALYSIS function and the following message will be displayed:

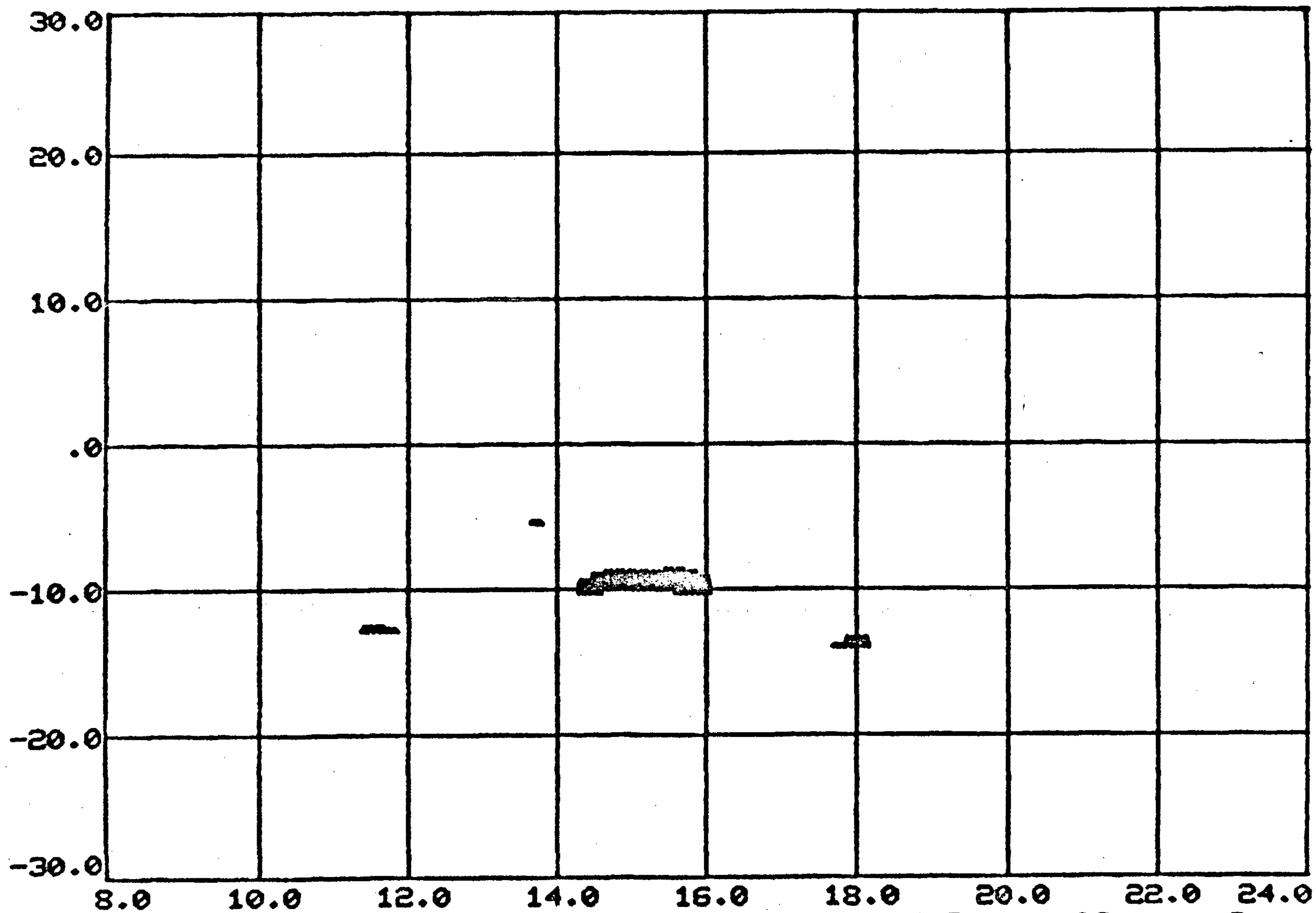
```
***** OLAAS DATA RECORDING & ANALYSIS  *****
           25 JANUARY 1978

      ** ENTER DESIRED FUNCTION **
1 = ON-LINE DATA RECORDING & ANALYSIS
2 = OFF-LINE ANALYSIS & PLOT DISPLAY
3 = DATE ENTRY OR CHANGE                             (    2)  1

***** ON-LINE DATA RECORDING & ANALYSIS *****

      **  ANSWER ANALYSIS QUESTIONS  **
SIGNAL MODE        CW = Ø, FM UP = 1, OR FM DOWN = 2 (    Ø)  Ø
```

2.3.6.4.1.1 Answer each question with the desired parameter required and the next question will appear. Figure 2.3.6.4-1 shows all possible "ON-LINE" questions that can be asked (Sense Switch 15 is on).

2.3.6.4.1.2 When the last question has been answered the raw data recording is done and then that data is analyzed and displayed. Figure 2.3.6.4-2 is an example of the analyzed display.

```
*****  OLAAS DATA RECORDING & ANALYSIS  *****
            25 JANUARY 1978

     ** ENTER DESIRED FUNCTION **
1 = ON-LINE DATA RECORDING & ANALYSIS
2 = OFF-LINE ANALYSIS & PLOT DISPLAY
3 = DATE ENTRY OR CHANGE                             (    2)  1

*****  ON-LINE DATA RECORDING & ANALYSIS  *****

     **  ANSWER ANALYSIS QUESTIONS  **
FREQUENCY          F1 = 1  OR  F2 = 2                (    1)
PULSE LENGTH       SHORT = 2  OR  LONG = 5           (    2)
SIGNAL MODE        CW = 0, FM UP = 1, OR FM DOWN = 2 (    0)
CW DOPPLER BAND    FULL=1, HALF=2, UP=3, OR DOWN=4   (    2)
CW NOTCH FILTER    WIDTH IN FREQUENCY CELLS  (0 TO 25) (   15)
START RANGE        IN KILOYARDS  (5 TO 35)           (    8)
STOP RANGE         IN KILOYARDS  (MAXIMUM 40)        (   24)
RANGE RESOLUTION   IN MULTIPLES OF 160 YARDS         (    1)
BEARING            ANGLE IN DEGREES (CENTER BEAM)    (    0)  2
BEAMS              NUMBER OF BEAMS                   (    3)  1
ANALYSIS THRESHOLD PERCENT OF AVERAGE POWER (0-10000) (  100)  2

     ** RAW DATA RECORDING QUESTIONS  **
START RANGE        IN KILOYARDS  (5 TO 35)           (    8)
STOP RANGE         IN KILOYARDS  (MAXIMUM 40)        (   24)
BEAMS              NUMBER OF BEAMS TO RECORD         (   16)  1
NUMBER PINGS       NUMBER OF PINGS TO RECORD         (    1)
CENTER BEAM        DESIRED CENTER BEAM DIRECTION     (    0)  2
DATA SOURCE        1=ACTIVE ON-LINE / 0=TAPE RECORDER (    1)  0
```

FIGURE 2.3.6.4-1




PING NO: 1 BEARING: 2 DOPPLER: HALF ANALYSIS THRESHOLD: 2x
SIGNAL: CW T1 F1 HANNING: 1 PLOT THRESHOLD: 1
TIME: 0:36:42 25 JANUARY 1978

Figure 2.3.6.4-2

2.3.6.4.1.3 At the CRT TERMINAL keyboard press RESET PAGE and the following message will be displayed:

```
NEW THRESHOLD PRECENTAGE FOR PLOT (1 - 1ØØ)
(1% DISPLAYS ALL ANALYZED DATA,  1ØØ% = NONE)
 TYPE Ø TO RETURN TO ANALYSIS PROGRAM
(     Ø)
```

2.3.6.4.1.4 The operator may desire to make a new plot using a higher "Plot Threshold", if so, enter the desired value (1 - 1ØØ) and a new plot will be displayed. This operation may be repeated as many times as desired. Enter a value of "Ø" to continue operation.

2.3.6.4.1.5 The RECORDED AND ANALYZED DATA STATUS TABLES will be displayed. Figure 2.3.6.4-3 is an example of the DATA STATUS TABLES. At the CRT TERMINAL keyboard press RESET PAGE and the following message will be displayed, the ON-LINE DATA RECORDING AND ANALYSIS is complete.

```
***** OLAAS DATA RECORDING & ANALYSIS *****
            25 JANUARY 1978

     ** ENTER DESIRED FUNCTION **
1 = ON-LINE DATA RECORDING & ANALYSIS
2 = OFF-LINE ANALYSIS & PLOT DISPLAY
3 = DATE ENTRY OR CHANGE                         (    1)
```

2.3.6.4.2 Off-Line Analysis and Plot Display.

2.3.6.4.2.1 Enter the value "2" for the OFF-LINE ANALYSIS AND PLOT DISPLAY function and the DATA STATUS TABLES will be displayed. Figure 2.3.6.4-3 is an example of the DATA STATUS TABLES.

2.3.6.4.2.2 At the CRT TERMINAL keyboard press RESET PAGE and the following message will be displayed:

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                25 JANUARY 1978

     ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY                            (    1)
```

2.3.6.4.2.2 Enter the indicated value for the desired off-line option and execute the procedure specified by the following paragraphs:

*********** OLAAS PROCESSING & DISPLAY SYSTEM ************
25 JANUARY 1978

RECORDED DATA STATUS TABLE

NUMBER OF PINGS RECORDED = 1
PING RECORD WINDOW = 7 TO 39 SECONDS
NUMBER OF BEAMS RECORDED = 16

| PING NUMBER | COMPASS (DEGREES) | FIRST BEAM | HEADING (DEGREES) | TIME |
|---|---|---|---|---|
| 1 | 346 | 11 | 211 | 8: 1: 4 |

ANALYZED DATA STATUS TABLE

| PING NUMBER | CENTER BEAM ANGLE | NUMBER BEAMS PROCESSED | NUMBER OF EXCEEDENCES | AVERAGE EXCEEDENCE |
|---|---|---|---|---|
| 1 | 31 | 1 | 366 | 25 |

FIGURE 2.3.6.4-3

```
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA     =  2.3.6.4.2.2.1
2 = INDIVIDUAL BEAM DISPLAY                  =  2.3.6.4.2.2.2
3 = MULTI-PING DISPLAY                       =  2.3.6.4.2.2.3
4 = FOUR PART DISPLAY                        =  2.3.6.4.2.2.4
```

2.3.6.4.2.2.1 Analysis of Previously Recorded Data Option.

2.3.6.4.2.2.1.1 Enter the value "1" for the ANALYSIS OF PREVIOUSLY RECORDED DATA option and the following message will be displayed:

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
               25 JANUARY 1978

     ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY                              (    1) 1

     **  ANSWER ANALYSIS QUESTIONS  **

FIRST PING        NUMBER FOR ANALYSIS (1 to 1Ø)    (    1)
```

2.3.6.4.2.2.1.2 Answer each question with the desired parameter required and the next question will appear. Figure 2.3.6.4-4 shows all possible "analysis" questions that can be asked (Sense Switch 15 is on).

2.3.6.4.2.2.1.3 When the last question has been answered the data is analyzed and displayed. Figure 2.3.6.4-2 is an example of the analyzed display.

2.3.6.4.2.2.1.4 At the CRT TERMINAL keyboard press RESET PAGE and the following message will be displayed:

```
 NEW THRESHOLD PERCENTAGE FOR PLOT (1 - 1ØØ)
 (1% DISPLAYS ALL ANALYZED DATA,  1ØØ% = NONE)
  TYPE Ø TO RETURN TO ANALYSIS PROGRAM
(    Ø)
```

2.3.6.4.2.2.1.5 The operator may desire to make a new plot using a higher "Plot Threshold", if so, enter the desired value (1 - 1ØØ) and a new plot will be displayed. This operation may be repeated as many times as desired.

2.3.6.4.2.2.1.6 The RECORDED AND ANALYZED DATA STATUS TABLES will be displayed. Figure 2.3.6.4-3 is an example of the DATA STATUS TABLES. At the CRT TERMINAL keyboard press RESET PAGE and the following message will be displayed, and the ANALYSIS OF PREVIOUSLY RECORDED DATA is complete.

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
               25 JANUARY 1978

     ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY                                    (    1)  1

     **  ANSWER ANALYSIS QUESTIONS  **

FIRST PING          NUMBER FOR ANALYSIS (1 TO 10)        (    1)
LAST PING           NUMBER FOR ANALYSIS (1 TO 10)        (    1)
FREQUENCY           F1 = 1  OR  F2 = 2                   (    1)
PULSE LENGTH        SHORT = 2  OR  LONG = 5              (    2)
SIGNAL MODE         CW = 0, FM UP = 1, OR FM DOWN = 2    (    0)
CW DOPPLER BAND     FULL=1, HALF=2, UP=3, OR DOWN=4      (    2)
CW NOTCH FILTER     WIDTH IN FREQUENCY CELLS  (0 TO 25)  (   15)
START RANGE         IN KILOYARDS  (5 TO 35)              (    8)
STOP RANGE          IN KILOYARDS  (MAXIMUM 40)           (   24)
RANGE RESOLUTION    IN MULTIPLES OF 160 YARDS            (    1)
BEARING             ANGLE IN DEGREES (CENTER BEAM)       (    0)
BEAMS               NUMBER OF BEAMS                      (    3)
ANALYSIS THRESHOLD  PERCENT OF AVERAGE POWER (0-10000)   (  100)
```

Figure 2.3.6.4-4

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                25 JANUARY 1978

     ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY                                  (    1)
```

2.3.6.4.2.2.2 Individual Beam Display Option.

2.3.6.4.2.2.2.1 Enter the value "2" for the INDIVIDUAL BEAM DISPLAY option and the following message will be displayed:

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                25 JANUARY 1978

     ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY                                  (    1)  2

     **  ANSWER INDIVIDUAL BEAM DISPLAY QUESTIONS **

DISPLAY PING    PING NUMBER TO BE DISPLAYED            (    1)
```

2.3.6.4.2.2.2.2 Answer each question with the desired parameter required and the next question will appear. Figure 2.3.6.4-5 shows all possible "INDIVIDUAL BEAM" questions that can be asked (Sense Switch 15 is on).

2.3.6.4.2.2.2.3 When the last question has been answered the raw data recording is done and then that data is analyzed and displayed. Figure 2.3.6.4-6 is an example of the individual beam display.

2.3.6.4.2.2.2.4 At the CRT TERMINAL keyboard press RESET PAGE and the following message will be displayed:

```
NEW THRESHOLD PERCENTAGE FOR PLOT (1 - 1ØØ)
(1% DISPLAYS ALL ANALYZED DATA,  1ØØ% = NONE)
 TYPE Ø TO RETURN TO ANALYSIS PROGRAM
(    Ø)
```

2.3.6.2.2.2.5 The operator may desire to make a new plot using a higher "Plot Threshold", if so, enter the desired value (1 - 1ØØ) and a new plot will be displayed. This operation may be repeated as many times as desired. Enter a value of "Ø" to continue operation.

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                29 JANUARY 1978

     ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY
                                                            (    5) 2

     **  ANSWER INDIVIDUAL BEAM DISPLAY QUESTIONS  **

DISPLAY PING        PING NUMBER TO BE DISPLAYED             (    1)
DISPLAY TYPE        2 = INDIVIDUAL   5 = BEAMS OR'ED        (    2)
START RANGE         IN KILOYARDS  (5 TO 35)                 (   20)
STOP RANGE          IN KILOYARDS  (MAXIMUM 40)              (   28)
BEARING             ANGLE IN DEGREES (CENTER BEAM)          (  338)
BEAMS               NUMBER OF BEAMS                         (    1)
PLOT THRESHOLD      1 TO 100 (%)  (1= DISPLAY ALL)          (    1)
```

Figure 2.3.6.4-5

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                  29 JANUARY 1978

    ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY
                                                  (     2)

    **  ANSWER INDIVIDUAL BEAM DISPLAY QUESTIONS  **

DISPLAY PING       PING NUMBER TO BE DISPLAYED       (     1)
DISPLAY TYPE       2 = INDIVIDUAL   5 = BEAMS OR'ED  (     2)  5
START RANGE        IN KILOYARDS  (5 TO 35)           (    20)
STOP RANGE         IN KILOYARDS  (MAXIMUM 40)        (    28)
BEARING            ANGLE IN DEGREES (CENTER BEAM)    (   338)
BEAMS              NUMBER OF BEAMS                   (     1)
PLOT THRESHOLD     1 TO 100 (%)  (1= DISPLAY ALL)    (     1)
```

Figure 2.3.6.4-6

2.3.6.4.2.2.2.6 The RECORDED AND ANALYZED DATA STATUS TABLES will be displayed. Figure 2.3.6.4-3 is an example of the DATA STATUS TABLES. At the CRT TERMINAL keyboard press RESET PAGE and the following message will be displayed, and the INDIVIDUAL BEAM DISPLAY is complete.

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                 25 JANUARY 1978

     ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY                                (    2)
```

2.3.6.4.2.2.2.7 The INDIVIDUAL BEAM DISPLAY contains an option for BEAMS OR'ED. To execute this option enter a value of "5" in answer to the DISPLAY TYPE question, all other operation are the same as paragraphs 2.3.6.4.2.2.2.1 through 2.3.6.4.2.2.2.6. Figures 2.3.6.4-7 and 2.3.6.4-8 are examples of the INDIVIDUAL BEAM DISPLAY question and BEAMS OR'ED display.

2.3.6.4.2.2.3 Multi-Ping Display Option.

2.3.6.4.2.2.3.1 Enter the value "3" for the MULTI-PING DISPLAY option and the following message will be displayed:

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                 25 JANUARY 1978

     ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY                                (    2) 3

     **  ANSWER MULTI-PING DISPLAY QUESTIONS  **

FIRST POINT       NUMBER FOR ANALYSIS (1 TO 1Ø)       (    1)
```

2.3.6.4.2.2.3.2 Answer each question with the desired parameter required and the next question will appear. Figure 2.3.6.4-9 shows all possible "MULTI-PING" questions that can be asked (Sense Switch 15 is on).

2.3.6.4.2.2.3.3 When the last question has been answered the data is displayed. Figure 2.3.6.4-10 is an example of the multi-ping display.

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                 25 JANUARY 1978

     ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY
                                                        (    2)  2

     **  ANSWER INDIVIDUAL BEAM DISPLAY QUESTIONS  **

DISPLAY PING        PING NUMBER TO BE DISPLAYED         (    1)
DISPLAY TYPE        2 = INDIVIDUAL   5 = BEAMS OR'ED    (    2)  5
BEARING             ANGLE IN DEGREES (CENTER BEAM)      (    2)
BEAMS               NUMBER OF BEAMS                     (    1)
PLOT THRESHOLD      1 TO 100 (%)  (1= DISPLAY ALL)      (    1)
```

Figure 2.3.6.4-7

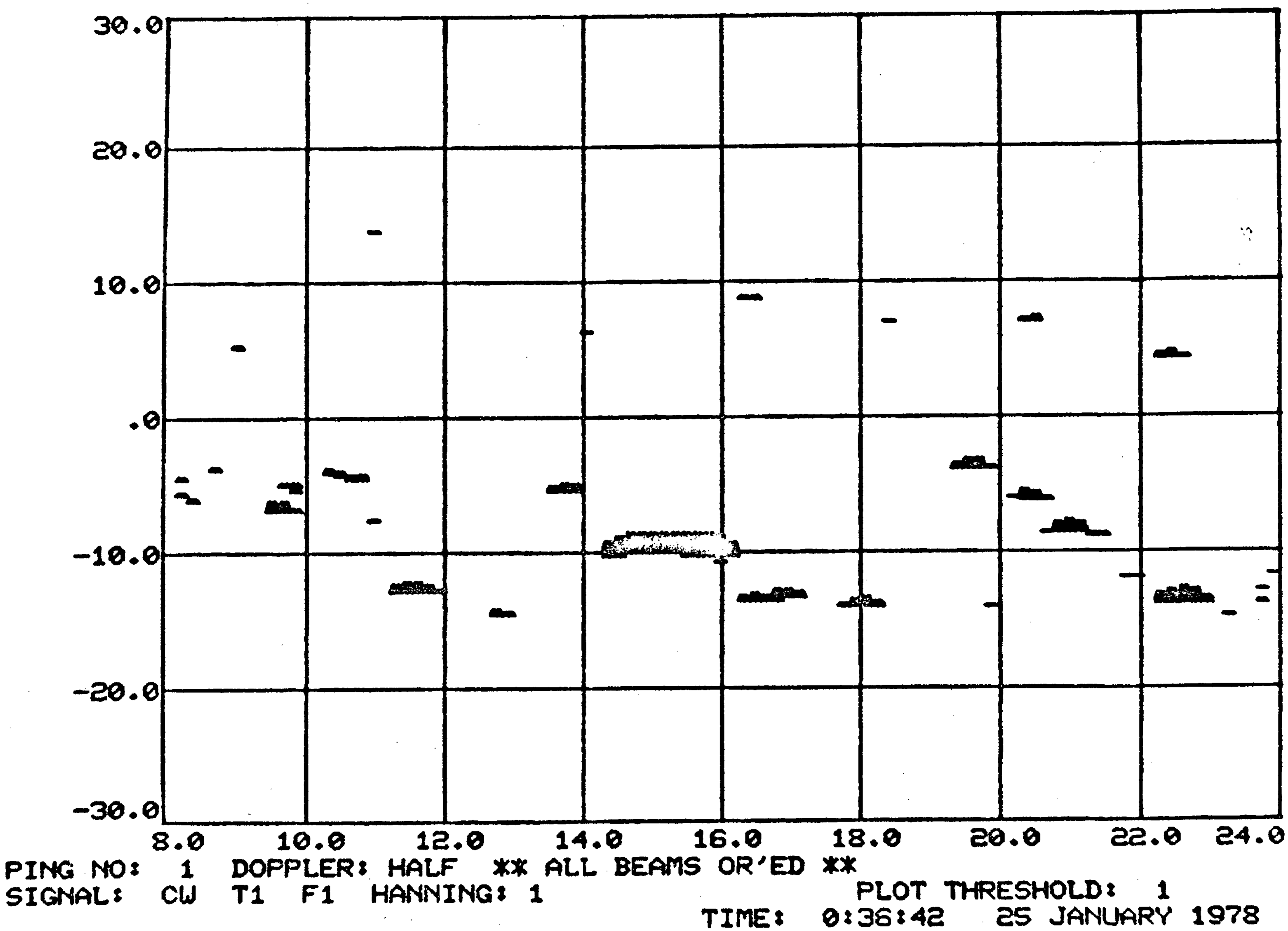



Figure 2.3.6.4-8

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                25 JANUARY 1978

     ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY                                   (    2) 3

     **  ANSWER MULTI-PING DISPLAY QUESTIONS  **

FIRST PING       NUMBER FOR ANALYSIS (1 TO 10)          (    1) 1
LAST PING        NUMBER FOR ANALYSIS (1 TO 10)          (    1) 1
START RANGE      IN KILOYARDS  (5 TO 35)                (    8) 8
STOP RANGE       IN KILOYARDS  (MAXIMUM 40)             (   24) 24
BEARING          ANGLE IN DEGREES (CENTER BEAM)         (    2) 2
PLOT THRESHOLD   1 TO 100 (%)  (1= DISPLAY ALL)         (    1) 1
```

Figure 2.3.6.4-9

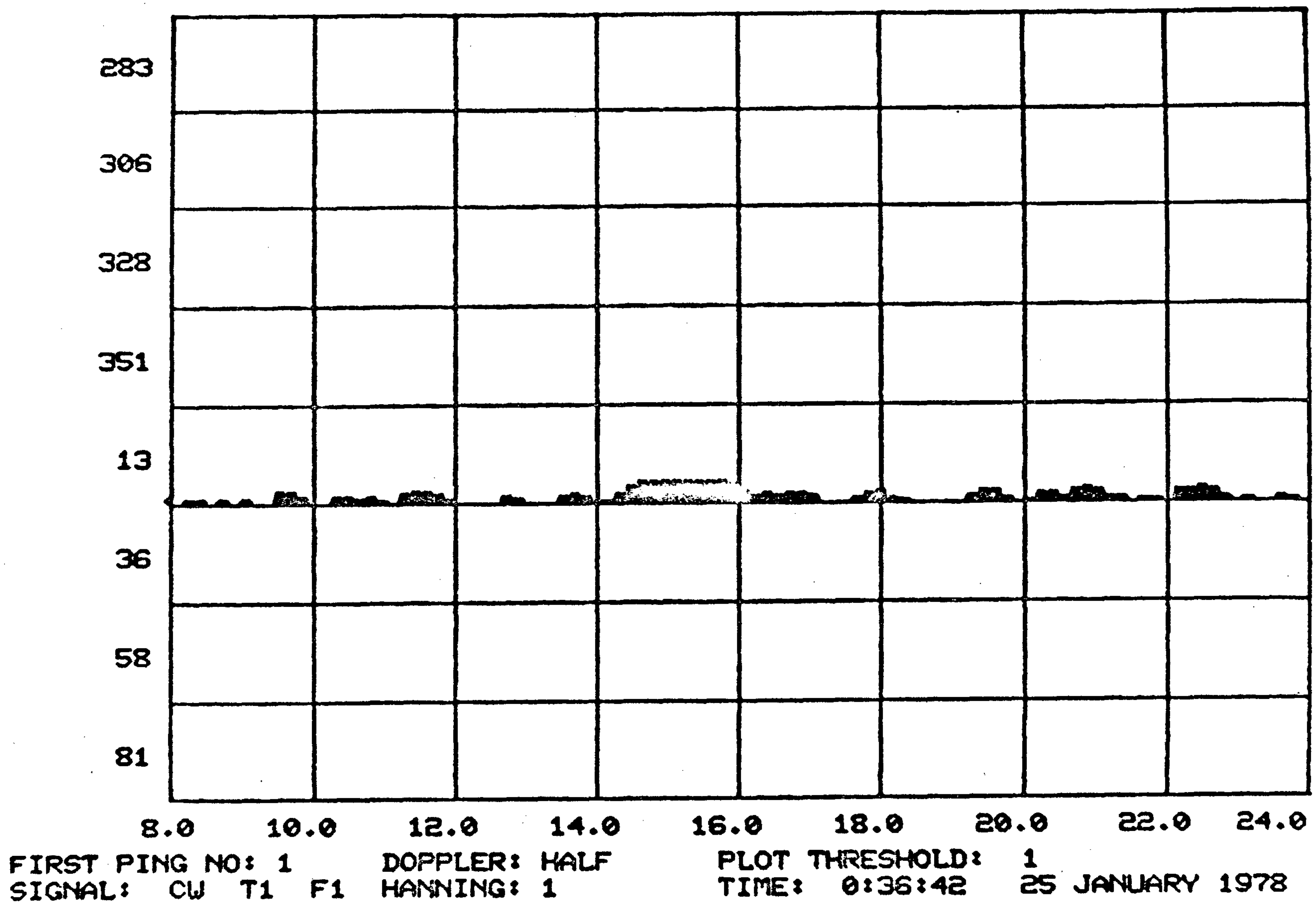



Figure 2.3.6.4-10

2.3.6.4.2.2.3.4 At the CRT TERMINAL keyboard press RESET PAGE and the following message will be displayed:

```
    ENTER NEW THRESHOLD PERCENTAGE FOR PLOT (1 - 1ØØ%)
     (1% DISPLAYS ALL ANALYZED AMPLITUDES, 1ØØ%= NONE)
     ENTER Ø TO RETURN TO ANALYSIS PROGRAM
  (    Ø)
```

2.3.6.2.2.3.5 The operator may desire to make a new plot using a higher "Plot Threshold", if so, enter the desired value (1 - 1ØØ and a new plot will be displayed. This operation may be repeated as many times as desired. Enter a value of "Ø" to continue operation.

2.3.6.4.2.2.3.6 The RECORDED AND ANALYZED DATA STATUS TABLES will be displayed. Figure 2.3.6.4-3 is an example of the DATA STATUS TABLES. At the CRT TERMINAL keyboard press RESET PAGE and the following message will be displayed, and the MULTI-PING DISPLAY is complete.

```
    ***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                   25 JANUARY 1978

         ** SELECT OFF-LINE OPTION **
    1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
    2 = INDIVIDUAL BEAM DISPLAY
    3 = MULTI-PING DISPLAY
    4 = FOUR PART DISPLAY                              (    3)
```

2.3.6.4.2.2.4 Four Part Display Option.

2.3.6.4.2.2.4.1 Enter the value "4" for the FOUR PART DISPLAY option and the following message will be displayed:

```
    ***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                   25 JANUARY 1978

         ** SELECT OFF-LINE OPTION **
    1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
    2 = INDIVIDUAL BEAM DISPLAY
    3 = MULTI-PING DISPLAY
    4 = FOUR PART DISPLAY                              (    3) 4

         **  ANSWER FOUR PART DISPLAY QUESTIONS  **

    DISPLAY PING      PING NUMBER TO BE DISPLAYED      (    1)
```

2.3.6.4.2.2.4.2 Answer each question with the desired parameter required and the next question will appear. Figure 2.3.6.4-11 shows all possible "ON-LINE" questions that can be asked (Sense Switch 15 is on).

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                25 JANUARY 1978

     ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY                                         (    3) 4

     **  ANSWER FOUR PART DISPLAY QUESTIONS  **

DISPLAY PING        PING NUMBER TO BE DISPLAYED               (    1) 1
DISPLAY BEARING     BEARING TO BE DISPLAYED (0 TO 359)        (    2) 2
START RANGE         IN KILOYARDS  (5 TO 35)                   (    8) 8
STOP RANGE          IN KILOYARDS  (MAXIMUM 40)                (   24) 24
PLOT THRESHOLD      1 TO 100 (%)  (1= DISPLAY ALL)            (    1) 1
```

Figure 2.3.6.4-11

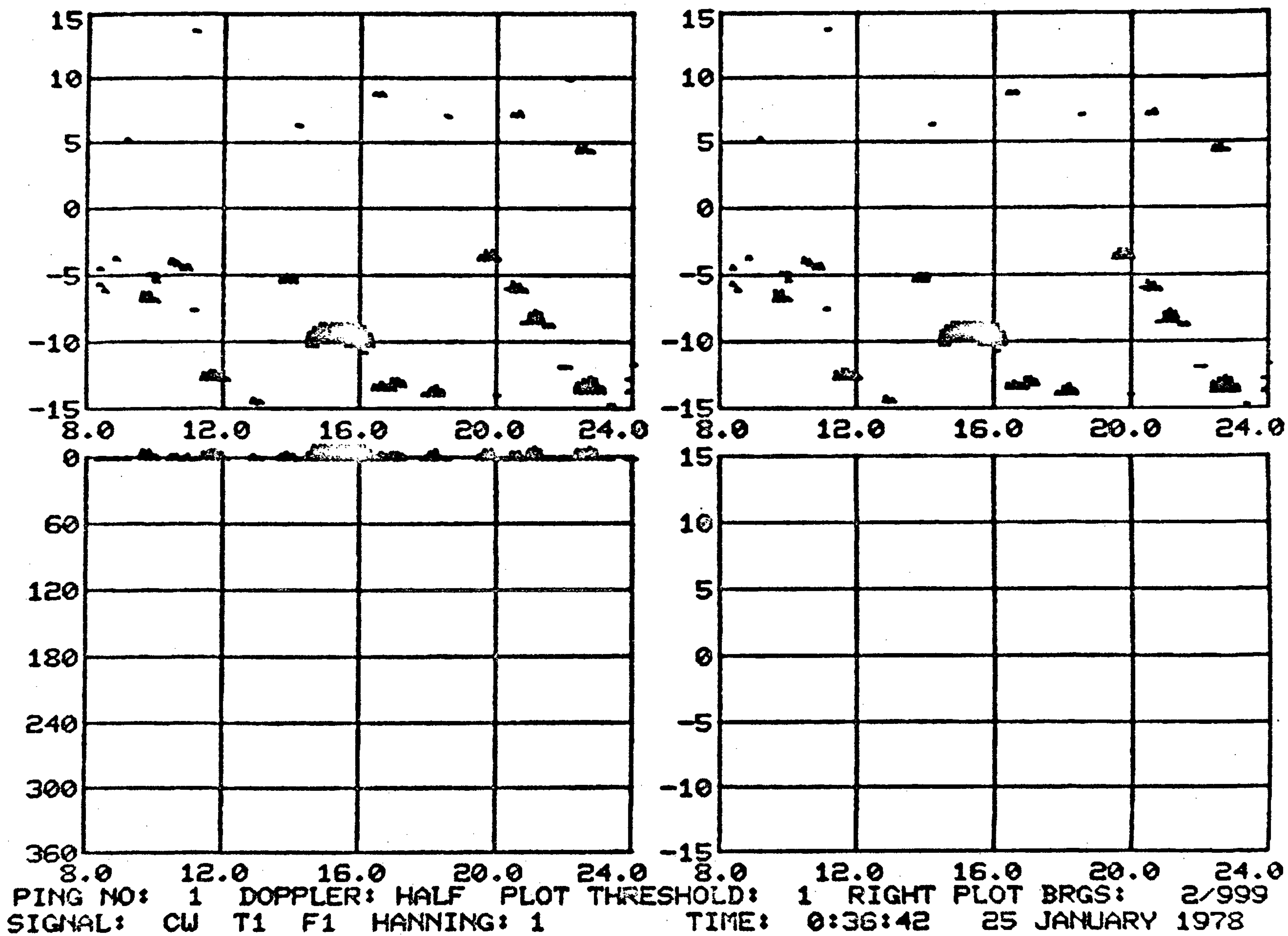



Figure 2.3.6.4-12

2.3.6.4.2.2.4.3 When the last question has been answered the data will be displayed. Figure 2.3.6.4-12 is an example of the FOUR PART DISPLAY.

2.3.6.4.2.2.4.4 At the CRT TERMINAL keyboard press RESET PAGE and the following message will be dispalyed:

```
NEW THRESHOLD PERCENTAGE FOR PLOT (1 - 1ØØ)
(1% DISPLAYS ALL ANALYZED DATA,  1ØØ% = NONE)
TYPE Ø TO RETURN TO ANALYSIS PROGRAM
(   Ø)
```

2.3.6.4.2.2.4.5 The operator may desire to make a new plot using a higher "Plot Threshold", if so, enter the desired value (1 - 1ØØ) and a new plot will be displayed. This operation may be repeated as many times as desired. Enter a value of "Ø" to continue operation.

2.3.6.4.2.2.4.6 The RECORDED AND ANALYZED DATA STATUS TABLES will be displayed. Figure 2.3.6.4-3 is an example of the DATA STATUS TABLES. At the CRT TERMINAL keyboard press RESET PAGE and the following message will be displayed, and the FOUR PART DISPLAY is complete.

```
***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY *****
                25 JANUARY 1978

     ** SELECT OFF-LINE OPTION **
1 = ANALYSIS OF PREVIOUSLY RECORDED DATA
2 = INDIVIDUAL BEAM DISPLAY
3 = MULTI-PING DISPLAY
4 = FOUR PART DISPLAY                                      (    4)
```

2.3.6.4.3 Date Entry or Change Function.

2.3.6.4.3.1 Enter the value "3" for the DATE ENTRY OR CHANGE function and the following message will be displayed:

```
*****  OLAAS DATA RECORDING & ANALYSIS  *****

     ** ENTER DESIRED FUNCTION **
1 = ON-LINE DATA RECORDING & ANALYSIS
2 = OFF-LINE ANALYSIS & PLOT DISPLAY
3 = DATE ENTRY OR CHANGE                                   (    2)  3

ENTER DATE
(                  )
```

If a date was entered in paragraph 2.3.6.3, then the following message will be displayed:

```
*****  OLAAS DATA RECORDING & ANALYSIS  *****
            25 JANUARY 1978

     ** ENTER DESIRED FUNCTION **
1 = ON-LINE DATA RECORDING & ANALYSIS
2 = OFF-LINE ANALYSIS & PLOT DISPLAY
3 = DATE ENTRY OR CHANGE                           (    2) 3

ENTER DATE
(25 JANUARY 1978   )
```

2.3.6.4.3.2 Enter the desired date and the following message will be displayed:

```
*****  OLAAS DATA RECORDING & ANALYSIS  *****
            25 JANUARY 1978

     ** ENTER DESIRED FUNCTION **
1 = ON-LINE DATE RECORDING & ANALYSIS
2 = OFF-LINE ANALYSIS & PLOT DISPLAY
3 = DATE ENTRY OR CHANGE                           (    2)
```

Note that if nothing was entered for the date then the date field is cleared.

2.3.7 User Program Analysis Operation.

2.3.7.1 Enter the letter "F" for the USER PROGRAM ANALYSIS operation and the following message will be displayed:

```
*****  OLAAS PROCESSING & DISPLAY SYSTEM  *****
               EXECUTIVE PROGRAM

         ** ENTER DESIRED OPERATION **
         D = DATA RECORDING & ANALYSIS
         F = USER PROGRAM ANALYZER
         G = FOURIER ANALYZER GRAPHICS

F
MAX BLOCKS #/SIZE/SPACE
       4/   2048/   8192
```

2.3.7.2 The operator may now call any of the available User Programs into execution. Figure 2.3.7-1 is an example of User Program Y7906 with the question asked in that program.

2.3.7.3 To return to the OLAAS EXECUTIVE PROGRAM, "re-boot" the system using OPERATING PROCEDURES: 2.4.1 - OLAAS System Loading Procedure.

```
*****  OLAAS PROCESSING & DISPLAY SYSTEM  *****
                EXECUTIVE PROGRAM

         ** ENTER DESIRED OPERATION **

         D = DATA RECORDING & ANALYSIS
         F = USER PROGRAM ANALYZER
         G = FOURIER ANALYZER GRAPHICS
F
MAX BLOCKS #/SIZE/SPACE
       4/   2048/   8192

Y 7906
PLEASE ENTER THE TYPE OF ANALYSIS DESIRED:
FOR STAVE INPUT, ENTER 1
FOR BEAM INPUT, ENTER 2
FOR ANALOG INPUT, ENTER 3
(     1)
1
ENTER THE STAVE NUMBER:
(     5)
16
PLEASE ENTER THE BLOCK SIZE DESIRED:
( 1024)
1024
ENTER THE NUMBER OF TIMES THAT YOU WANT ME TO
AVERAGE THE INPUT:
(    10)
15
HOW MANY HANNINGS WITHIN EACH AVERAGING LOOP??
(     1)
1
WORKING.......
```

Figure 2.3.7-1

2.3.8 Fourier Analyzer Graphics Operation.

2.3.8.1 Enter the letter "G" for the FOURIER ANALYZER GRAPHICS operation and the following message is all that is displayed:

```
***** OLAAS PROCESSING & DISPLAY SYSTEM *****
            EXECUTIVE PROGRAM

         ** ENTER DESIRED OPERATION **

         D = DATA RECORDING & ANALYSIS
         F = USER PROGRAM ANALYZER
         G = FOURIER ANALYZER GRAPHICS
G
```

2.3.8.2 The operator will enter at the Fourier Analyzer keyboard a JUMP and Ø, then press RETURN. The contents of Fourier Data Block Ø will be displayed on the screen. Note: Fourier Data Block Ø must be loaded prior to calling the Fourier Analyzer Graphics.

2.3.8.2 To return to the OLAAS EXECUTIVE Program, "re-boot" the system using OPERATING PROCEDURE:

2.3.9 Error Halts. During exeuction of the data collection program, various error conditions will cause the program to halt. When the computer halts, the display shows the halt instruction. These halts and their meanings are as follows:

102002 Beam former completion (DMA) interrupt did not occur within 1.2 seconds after completion of disc write.

102003 Beam former completion (DMA) interrupt waiting when disc write completed.

102004 A Time Base Generator tick was missed (irrecoverable).

102005 Processing between transmit gate and start of recording took too long.

102006 Disc write length not a multiple of 128 words.

102007 Disc storage area exceeded.

102010 Beam former input length request not a multiple of 16 words.

102012 No disc space allowed.

102013 No 2 consecutive clock samples alike out of 4.

102030 Beam former interrupt occurred.

## 2.4 BOOTSTRAP LOADING PROCEDURES

The following BOOTSTRAP LOADING PROCEDURES are used to load the OLAAS PROCESSING AND DISPLAY SYSTEM, make changes to, and generate the system:

2.4.1 OLAAS System Loading.

2.4.2 Read Time Executive (RTE) System Loading.

2.4.3 Fourier (BCS) System Loading.

2.4.4 Paper Tape Loading.

2.4.5 Writing to Disc.

2.4.6 Reloading HP-2100 Computer System Loader.

OPERATING PROCEDURE: 2.4.1 - OLAAS SYSTEM LOADING PROCEDURE

The OLAAS SYSTEM LOADING PROCEDURE will "Bootstrap", or load in the OLAAS SYSTEM and start execution.

| STEP NO. | DESCRIPTION | OBSERVATION |
|---|---|---|
| | At the HP-2100S COMPUTER OPERATOR PANEL | |
| 1. | Set P = ∅7775∅ | DISPLAY REGISTER = ∅7775∅ |
| 2. | Press S and then press CLEAR DISPLAY | DISPLAY REGISTER = ∅∅∅∅∅∅ |
| 3. | Press EXTERNAL PRESET and INTERNAL PRESET | |
| 4. | Press LOADER ENABLE | |
| 5. | Press HALT/CYCLE | |
| 6. | Press MEMORY DATA and clear BIT 9 of the DISPLAY REGISTER if set | DISPLAY REGISTER = ∅3∅∅∅∅ |
| 7. | Press RUN twice | |
| 8. | Set P = ∅∅∅∅∅2 | DISPLAY REGISTER = ∅∅∅∅∅2 |
| 9. | Press RUN | The following message is displayed on the CRT Terminal screen:<br>OLAAS PROCESSING & DISPLAY SYSTEM<br>EXECUTIVE PROGRAM<br>** ENTER DESIRED OPERATION **<br>D = DATA RECORDING & ANALYSIS<br>F = USER PROGRAM ANALYZER<br>G = FOURIER ANALYZER GRAPHICS |
| | THE OLAAS SYSTEM LOAD IS COMPLETE | |

OPERATING PROCEDURE: 2.4.2 - REAL TIME EXECUTIVE (RTE) SYSTEM LOADING PROCEDURE

The REAL TIME EXECUTIVE (RTE) SYSTEM LOADING PROCEDURE will "Bootstrap", or load in, the RTE SYSTEM. The RTE SYSTEM is used with the FILE MANAGEMENT (FMGR) program for program creation and changes, also generating Fourier coreloads.

| STEP NO. | DESCRIPTION | OBSERVATION |
|---|---|---|
| | At the HP-2100S COMPUTER OPERATOR PANEL | |
| 1. | Set P = Ø7775Ø | DISPLAY REGISTER = Ø7775Ø |
| 2. | Press S and then press CLEAR DISPLAY | DISPLAY REGISTER = ØØØØØØ |
| 3. | Press EXTERNAL PRESET and INTERNAL PRESET | |
| 4. | Press LOADER ENABLE | |
| 5. | Press HALT/CYCLE | |
| 6. | Press MEMORY DATA and set BIT 9 of the DISPLAY REGISTER | DISPLAY REGISTER = Ø31ØØØ |
| 7. | Press RUN twice | The following message is displayed on the CRT Terminal screen:<br>SET TIME |
| | THE RTE SYSTEM LOAD IS COMPLETE | |
| | At the CRT TERMINAL KEYBOARD | |
| 8. | Press the SPACE BAR | The CRT TERMINAL screen displays:<br>* |
| 9. | Key in: RU,FMGR and then press RETURN | The CRT TERMINAL screen displays:<br>*RU,FMGR<br>(Note: The "prompt" on the CRT TERMINAL sceeen for FMGR is a colon (:). |
| | THE FILE MANAGEMENT PROGRAM IS LOADED AND EXECUTING | |

OPERATING PROCEDURE: 2.4.3 - FOURIER BASIC CONTROL SYSTEM (BCS) LOADING PROCEDURE

The FOURIER BASIC CONTROL SYSTEM (BCS) LOADING PROCEDURE will "Bootstrap", or load in, the FOURIER (BCS) SYSTEM and start execution at the selected Coreload starting cylinder on the Fourier Disc. The FOURIER (BCS) SYSTEM is used to start the system for a given coreload so that a Fourier Analyzer Program (Keyboard Stack) may be written or execution of the coreload.

| STEP NO. | DESCRIPTION | OBSERVATION |
|---|---|---|
| | At the HP-2100S COMPUTER OPERATOR PANEL | |
| 1. | Set P = Ø7775Ø | DISPLAY REGISTER = Ø7775Ø |
| 2. | Press S and then press CLEAR DISPLAY | DISPLAY REGISTER = ØØØØØØ |
| 3. | Press EXTERNAL PRESET and INTERNAL PRESET | |
| 4. | Press LOADER ENABLE | |
| 5. | PRESS HALT/CYCLE | |
| 6. | Press MEMORY DATA and clear BIT 9 of the DISPLAY REGISTER if set | DISPLAY REGISTER = Ø3ØØØØ |
| 7. | Press RUN | |
| 8. | Press B and then set the DISPLAY REGISTER to the starting cylinder number (in octal) | DISPLAY REGISTER = Cylinder Number (in octal)<br>Coreload Ø = 1<br>Coreload 1 = 7<br>Coreload 2 = 15<br>Coreload 3 = 23<br>Coreload 4 = 31<br>Coreload 5 = 37<br>Coreload 6 = 45<br>Coreload 7 = 53 |
| 9. | Press RUN | |
| 10. | Set P = ØØØØØ2 | DISPLAY REGISTER = ØØØØØ2 |
| 11. | Press RUN | |
| | THE FOURIER (BCS) SYSTEM LOAD IS COMPLETE | |

OPERATING PROCEDURE: 2.4.4 - PAPER TAPE LOADING PROCEDURE

The PAPER TAPE LOADING PROCEDURE will cause a paper tape to be read in and stored in core memory. The input program shall be in binary form, containing absolute addresses.

| STEP NO. | DESCRIPTION | OBSERVATION |
|---|---|---|
| | AT THE PAPER TAPE READER | |
| 1. | Turn on the PAPER TAPE READER | Photoreader is illuminated |
| 2. | Press LOAD and place paper tape in reader | Ensure tape is not installed backwards. |
| 3. | Press READ | |
| | At The HP-2100S COMPUTER OPERATOR PANEL | |
| 4. | Set P = Ø777ØØ | DISPLAY REGISTER = Ø777ØØ |
| 5. | Press EXTERNAL PRESET and INTERNAL PRESET | |
| 6. | Press LOADER ENABLE | |
| 7. | Press RUN | The tape should now read in, and stop when complete. DISPLAY REGISTER = 1Ø2Ø77 (Note: Halt code = 1Ø2Ø55 is an address error and halt code = 1Ø2Ø11 is a checksum error.) |

THE LOAD IS COMPLETE

OPERATING PROCEDURE: 2.4.5 - WRITING TO DISC PROCEDURE

The WRITING TO DISC PROCEDURE is used for dumping core to a disc pack. In the OLAAS System this procedure is used to write the Fourier Analyzer Programs (Keyboard Stack) into the desired coreload so that the Keyboard Stack will be executed when the coreload in entered.

| STEP NO. | DESCRIPTION | OBSERVATION |
|---|---|---|
| | At the HP-2100S COMPUTER OPERATOR PANEL | |
| 1. | Set P = ∅776∅∅ | DISPLAY REGISTER = ∅776∅∅ |
| 2. | Press A and then press CLEAR DISPLAY | DISPLAY REGISTER = ∅∅∅∅∅∅ |
| 3. | Press B and then set the DISPLAY REGISTER to the starting cylinder number (in octal), also set BIT 15 | DISPLAY REGISTER = 1∅∅∅xx<br>xx = Cylinder Number (in octal)<br>Coreload ∅ = ∅1<br>Coreload 1 = ∅7<br>Coreload 2 = 15<br>Coreload 3 = 23<br>Coreload 4 = 31<br>Coreload 5 = 37<br>Coreload 6 = 45<br>Coreload 7 = 53 |
| 4. | Press EXTERNAL PRESET and INTERNAL PRESET | |
| 5. | Press RUN | The Computer will HALT and the DISPLAY REGISTER = 1∅2∅77.<br>(Note: If the correct HALT is not obtained, repeat the procedure.) |
| | WRITING TO DISC IS COMPLETE | |

OPERATING PROCEDURE: 2.4.6 - RELOADING HP-21ØØS COMPUTER SYSTEM LOADER PROCEDURE

The RELOADING HP-21ØØS COMPUTER SYSTEM LOADER PROCEDURE will re-install the System Loader if it has been cleared out of core memory. The System Loader is located in upper core and should never be reached, however, if it is cleared and Procedures 2.4.1 or 2.4.2 can not be executed, then re-install the System Loader using this procedure

| STEP NO. | DESCRIPTION | OBSERVATION |
|---|---|---|
| | At the HP-21ØØS COMPUTER OPERATOR PANEL | |
| 1. | Press HALT/CYCLE | HALT/CYCLE is illuminated |
| 2. | Set M = ØØØØ2Ø | DISPLAY REGISTER = ØØØØ2Ø |
| 3. | Press MEMORY DATA and then enter the following numbers (in octal) press INCREMENT M after each number except the last number | DISPLAY REGISTER = Number Entered |
| | 1Ø371Ø | |
| | 1Ø231Ø | |
| | Ø24Ø21 | |
| | 1Ø251Ø | |
| | ØØ1727 | |
| | 1Ø371Ø | |
| | 1Ø231Ø | |
| | Ø24Ø26 | |
| | 1Ø241Ø | |
| | 17ØØØ1 | |
| | ØØ6ØØ4 | |
| | Ø24Ø2Ø (last number) | |
| 4. | After entering last number (Ø24Ø2Ø), then set B = Ø777ØØ | DISPLAY REGISTER = Ø777ØØ |
| 5. | Set P = ØØØØ2Ø | DISPLAY REGISTER = ØØØØ2Ø |
| 6. | Press EXTERNAL PRESET and INTERNAL PRESET | |
| | Continue on Page 2 | |

OPERATING PROCEDURE: 2.4.6 CONTINUED - RELOADING HP-21ØØS COMPUTER SYSTEM LOADER PROCEDURE

| STEP NO. | DESCRIPTION | OBSERVATION |
|---|---|---|
| | At the PAPER TAPE READER | |
| 7. | Turn on the PAPER TAPE READER | Photoreader is illuminated |
| 8. | Press LOAD and place paper tape P/N Ø5451-9ØØØ7 in reader | Ensure tape is not installed backwards |
| 9. | Press READ | |
| | At the HP21ØØS COMPUTER OPERATOR PANEL | |
| 1Ø. | Press LOADER ENABLE | LOADER ENABLE is illuminated |
| 11. | Press RUN | The tape should now read in, and stop when complete. |
| 12. | Press HALT/CYCLE | HALT/CYCLE is illuminated |
| | NOTE: Since the bootstrap tape is for a 16K system, the following changes must be made. | |
| 13. | Press LOADER ENABLE | LOADER ENABLE is illuminated |
| 14. | Set M = Ø77772 | DISPLAY REGISTER = Ø77772 |
| 15. | Set MEMORY DATA = 1ØØ1ØØ | DISPLAY REGISTER = 1ØØ1ØØ |
| 16. | Press LOADER ENABLE | LOADER ENABLE not illuminated |
| | NOTE: If the following tape does not load in, repeat steps 1 -16. | |
| | At the PAPER TAPE READER | |
| 17. | Press LOAD and place paper tape "EMMETT IPDBL" in reader | Ensure tape is not installed backwards. |
| 18. | Press READ | |
| | At the HP-21ØØS COMPUTER OPERATOR PANEL | |
| 19. | Set P = Ø777ØØ | DISPLAY REGISTER = Ø777ØØ |
| 2Ø. | Press EXTERNAL PRESET and INTERNAL PRESET | |
| 21. | Press LOADER ENABLE | LOADER ENABLE is illuminated |
| 22. | Press RUN | The tape should now read in, and stop when complete. (Repeat steps 1-16 if tape does not load) |
| | Continue on Page 3 | |

OPERATING PROCEDURE: 2.4.6 CONTINUED - RELOADING HP-21ØØS COMPUTER SYSTEM LOADER PROCEDURE

| STEP NO. | DESCRIPTION | OBSERVATION |
|---|---|---|
| 23. | Set S = ØØ171Ø | DISPLAY REGISTER = ØØ171Ø |
| 24. | Set P = ØØØØØ2 | DISPLAY REGISTER = ØØØØØ2 |
| 25. | Press EXTERNAL PRESET and INTERNAL PRESET | |
| 26. | Press LOADER ENABLE | LOADER ENABLE is illuminated |
| 27. | Press RUN | The Computer will HALT with the DISPLAY REGISTER = 1Ø2Ø77, which signifies PDBBL is installed. |
| | At the PAPER TAPE READER | |
| 28. | Press LOAD and place the paper tape "FBOOT", P/N Ø5451-1Ø98382, in reader; then position the tape to the second program on the tape | Ensure tape is not installed backwards |
| | At the HP-21ØØS COMPUTER OPERATOR PANEL | |
| 29. | Set P = Ø777ØØ | DISPLAY REGISTER = Ø777ØØ |
| 3Ø. | Press EXTERNAL PRESET and INTERNAL PRESET | |
| 31. | Press LOADER ENABLE | LOADER ENABLE is illuminated |
| 32. | Press RUN | The tape should now read in, and stop before finishing the tape |
| 33. | Set S = 1776ØØ | DISPLAY REGISTER = 1776ØØ |
| 34. | Set P = ØØØØØ2 | DISPLAY REGISTER = ØØØØØ2 |
| 35. | Press EXTERNAL PRESET and INTERNAL PRESET | |
| 36. | Press RUN | The remaining tape should now read in, and stop when complete. DISPLAY REGISTER = 1Ø2Ø76 which indicates a successful load. |
| | THE SYSTEM SHOULD NOW BE OPERATIONAL | |

## 2.5 CORE LOAD GENERATION PROCEDURES

The Fourier disc is composed of eight core loads, as follows:

0 OLAAS Executive

1 OLAAS Data Collection, Analysis and Executive

2 OLAAS Raw Data Collection

3 Reserved for swapping CL 1

4 Reserved for swapping CL 2

5 OLAAS User Programs

6 Fourier Analyzer Graphics

7 OLAAS Plot Display

Each of these core loads is built using the Fourier Cross Loader (FXL) and the File Copy Program (COPYF), operating under RTE. For each core load there is a separate command file and a transfer file, both of which control FXL. The command file is read by FXL and controls its functioning. The transfer file starts the operation of FXL. All of these files may be edited with the Interactive Editor. The Fourier core loads require the file MPSYS, containing the BCS system, be present. To modify the BCS system, it is necessary to modify transfer file MPTR, FXL command file MPCMCS, and/or BCS generation command file MPPC, and transfer control to MPTR. Table 2.5-1 lists the files accessed in generating a replacement core load.

To illustrate the process of core load generation task, for example, the case of a modification to the executive. The steps would be as follows:

1. Boot RTE
2. RU, FMGR
3. RU, EDITR
4. Make necessary modifications to the executive program source file, MPØØ3. Leave edited file in MPØØ3 and logical source tracks by terminating EDITR with command ELR.
5. LG,2
6. RU,FTN4,2,99 (assemble new program)
7. PU,YØØØ3 (Purge old relocatable file)
8. SA,LG,#YØØØ3::::-1 (Save new relocatable file)
9. TR,MPTRØ (Create file containing #YØØØ3 and subroutines)
10. TR,MPDCP (Run FXL, creating absolute file DCPSYS)

11. RU,COPYF (Run File Copy program)

12. Answer COPYF questions as follows:

    CORE SIZE: 32

    CORELOAD NUMBER: Ø (for this example)

    DISC LU NUMBER: 9

    SYSTEM FILE NAME: DCPSYS

    OVERLAY FILE USED: NO

    PROCEED: YES

## 2.6 FOURIER ANALYZER PROGRAM (KEYBOARD STACK)

The Fourier Analyzer Program Keyboard Stack, is a sequence of commands that the Fourier Analyzer will perform automatically. Reference the Fourier Analyzer System - Operating Manual, 04541-90199 with supplements, Section VII for a complete description of this type of program.

In the OLAAS system, the USTRT subroutine transfers control to Label Ø (zero) in the keyboard program stack. The way that the analysis program, or any other program in any other core load, is started immediately, rather than requiring operator action, is that the keyboard program stack contains the program:

```
L    0
Y    0002 (for example)
.
```

This must be entered and stored on the disc whenever the core load in question is regenerated. To do so, follow this procedure:

1. Load the core load in question using OPERATING PROCEDURE: 2.4.3 - Fourier (BCS) System Loading.

2. At the Fourier Analyzer keyboard, press:

   REPLACE, ENTER
   LABEL 0, ENTER
   USER PROG 0002, ENTER
   END, ENTER
   TERM, ENTER

3. Press L, ENTER (the program stack will be displayed).

4. At the computer, write the program stack to disc using OPERATING PROCEDURE: 2.4.5 - Writing to DISC.

## OLAAS PROCESSING & DISPLAY SYSTEM

### CORELOAD GENERATION FILE NAMES

| CORELOADS: | CL Ø | CL 1 | CL 2 | CL 5 | CL 6 | CL 7 |
|---|---|---|---|---|---|---|
| MAIN PROGRAM SOURCE FILE: | MPØØ3A | MPØØ2A | YØØØ1 | (Fourier System) | (Fourier System) | MPØØ7A |
| MAIN PROGRAM OBJECT FILE: | #YØØØ3 | Y1ØØ20 | #YØØØ1 | (Fourier System) | (Fourier System) | Y1ØØ7A |
| BCS GENERATION COMMAND FILE: | RTPC | N/A | RTPC | N/A | N/A | N/A |
| FXL COMMAND FILE: | RTCM | MPCM1 | RTCM | MPCM5 | MPCM6 | MPCM7 |
| FXL TRANSFER FILE: | MPTRØ | MPTR1 | MPTR2 | MPTR5 | MPTR6 | MPTR7 |
| SYSTEM FILE FOR "COPYF": | DCPSYS | MPSYS | DCPSYS | MPSYS | MPSYS | MPSYS |
| OVERLAY FILE: | N/A | RAF | N/A | RAF | RAF | RAF |
| SPECIAL GENERATION PROCEDURES: | TR,MPTRØ<br>TR,RTTR | TR,MPTR1 | TR,DCPTR<br>TR,RTTR | TR,MPTR5 | TR,MPTR6 | TR,MPTR7 |

<u>BCS SYSTEM GENERATION:</u> MPSYS

FXL TRANSFER FILE: MPTR

FXL COMMAND FILE: BR1CSC

BCS COMMAND FILE: RAFPC

SPECIAL GENERATION PROCEDURE: TR,MPTR

<u>DISC FORMATTING PROCEDURE:</u>

RU,FMTXX1

Figure 2.5-1

## 2.7 DISC COPY - MADIU PROGRAM

DISC COPY uses the MADIU Program which is a Hewlett Packard Magnetic Tape/ Disc Utility program for copying disc images. MADIU is fully described in the Magnetic Tape/Disc Utility Installation/Operation Note, 0541-90307. The OPERATING PROCEDURE: 2.7 - Disc Copy Procedure (MADIU Program) will execute the MADIU program in the OLAAS System environment.

The following messages is an example of a disc copy from the lower subchannel, Ø, to upper subchannel, 1.

```
(FOR 79ØØ) TYPE "LO" "DU" "VE" "CO" OR "TE"
OO
   SOURCE SUBCHANNEL # ?
Ø
 "DESTINATION SUBCHANNEL # ?"
1
```

OPERATING PROCEDURE: 2.7 - DISC COPY PROCEDURE (MADIU PROGRAM)

The DISC COPY PROCEDURE uses the MADIU Program, a HP Mag Tape/Disc Utility program for copying disc images. MADIU is fully described in the Magnetic Tape/Disc Utility Installation/Operation Note, Ø5451-9Ø3Ø7, (see 4.0 References). The version used with the Fourier Bootstrap Software is configured with 4k SIO driver and no MAG TAPE capability so that is will fit on disc cylinder Ø. Other operational caracteristics are the same.

| STEP NO. | DESCRIPTION | OBSERVATION |
|---|---|---|
| | At the PAPER TAPE READER | |
| 1. | Load the unconfigured MADIU binary paper tape using OPERATING PROCEDURE 2.4.4 - Paper Tape Loading | The MADIU Program is successfully loaded |
| | At the HP-21ØØS COMPUTER OPERATOR PANEL | |
| 2. | Set P = ØØØ12Ø | DISPLAY REGISTER = ØØØ12Ø |
| 3. | Press EXTERNAL PRESET and INTERNAL PRESET | |
| 4. | Press RUN | Answer the questions on the CRT Terminal screen |
| | At the CRT TERMINAL | |
| 5. | Answer the question of what operation to perform = CO (CO - Copy disc to disc (with auto verify)) | The CRT TERMINAL screen displays: TYPE "LO" "DU" "VE" "CO" OR "TE" |
| 6. | Press LF | Next question is displayed |
| 7. | Answer the question of which subchannel contains the "source" to be copied. (NOTE: The FIXED PLATTER is Ø and the REMOVABLE PLATTER is 1 ) | The CRT TERMINAL screen displays: SOURCE SUBCHANNEL # ? |
| 8. | Press LF | Next question is displayed |
| 9. | Answer the question of which subchannel is to be the "destination" of the copy | The CRT TERMINAL screen displays: DESTINATION SUBCHANNEL # ? |
| 1Ø. | Turn the PROTECT/OVERRIDE switch to "override" (NOTE: The PROTECT/OVERRIDE switch is located on the 79ØØ DISC Unit behind the panel) | |
| 11. | Press LF | DISPLAY REGISTER = Count up from ØØØØØ1 to ØØØ312 |
| 12. | Turn the PROTECT/OVERRIDE switch to "protect" | Any error displayed may be found in the MADIU Operation Note |
| | DISC TO DISC COPY IS COMPLETE | |

3.0 OLAAS DISC STORAGE

3.1 INTRODUCTION

The OLAAS Disc Storage utilizes the Hewlett Packard 79ØØA Cartridge Disc Drive as the mass store device on the Fourier System. The HP79ØØA disc has an upper (removable cartridge) and lower (fixed) discs. The OLAAS system utilizes both discs. The lower (fixed) disc contains the RTE Software System, OLAAS source and object program files, and system generation files. See Appendix C for the RTE System Disc Directory. The upper (removable cartridge) disc contains the OLAAS Processing and Display System, referred to as the OLAAS Fourier Disc.

3.2 OLAAS FOURIER DISC LAYOUT

The OLAAS Fourier Disc layout is defined by the FMTXX1 program, that has been configured for this system, and is executed when the basic system (MPSYS) is generated. See Appendix B for the FMTXX1 program listing. Figure 3.2-1 is the OLAAS Fourier Disc configuration.

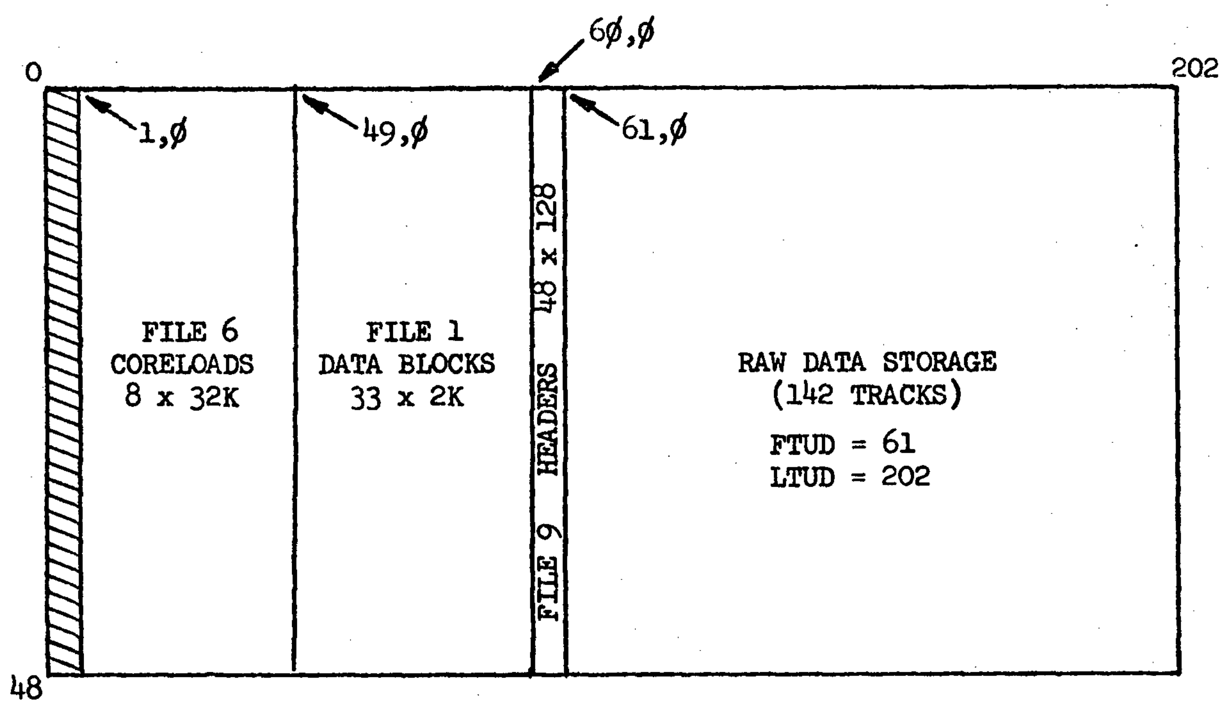



FIGURE 3.2-1 OLAAS Fourier Disc Configuration

### 3.2.1 FILE STRUCTURE

FILE 1 - DATA BLOCK

File 1, Disc Data Block, is set up for storage of data. Data block Ø is used for the system common data vector, ICOM. Each ping has reserved three (3) data blocks with the capacity for storaging 1536 data points for each ping, with a maximum of ten (1Ø) pings (data blocks 1-3Ø) with 31 and 32 not used.

During analysis of raw data recorded on the disc, each amplitude is tested against a threshold. For each point which exceeds the analysis threshold, four integers are saved in the data blocks: 1) beam direction, 2) range index, 3) compressed amplitude and 4) doppler bin.

FILE 6 - SYSTEM CORELOAD

File 6, System Coreload, contains core images of the eight OLAAS coreloads. Using core swapping, the various coreloads are brought into the system for execution or returned to the disc. The OLAAS coreloads are as follows:

| Coreload | Function |
|---|---|
| Ø | OLAAS Executive Program |
| 1 | OLAAS Analysis & System Executive Subroutine |
| 2 | OLAAS Raw Data Collection Subroutine |
| 3 | Copy of coreload 1 (used for processing) |
| 4 | Copy of coreload 2 (used for multi-ping recording) |
| 5 | OLAAS User Programs Analysis |
| 6 | Fourier Analyzer Graphics |
| 7 | OLAAS Plot Display Subroutine |

FILE 9 - HEADERS

File 9, Headers, are associated with the data blocks and contain the data qualifiers.

### 3.2.2 RAW DATA STORAGE

Disc tracks 61 through 202, 142 tracks, are used for Raw Data storage (recording) of Buoy Data. The actual amount of data recorded is dependent upon number of beams, pings, and recording time.

## 3.3 OLAAS FOURIER DISC GENERATION

The generation of the OLAAS Fourier Disc is the result of creating the various coreloads. The system configuration is used when the coreloads are generated and Operating Procedure: 2.5 - Coreload Generation Procedure, when executed will create a coreload. Table 3.3-1 shows the files used. to generate the coreloads.

## OLAAS PROCESSING & DISPLAY SYSTEM

### CORELOAD GENERATION FILE NAMES

| CORELOADS: | CL Ø | CL 1 | CL 2 | CL 5 | CL 6 | CL 7 |
|---|---|---|---|---|---|---|
| MAIN PROGRAM SOURCE FILE: | MPØØ3A | MPØØ2A | YØØØ1 | (Fourier System) | (Fourier System) | MPØØ7A |
| MAIN PROGRAM OBJECT FILE: | #YØØØ3 | Y1ØØ20 | #YØØØ1 | (Fourier System) | (Fourier System) | Y1ØØ7A |
| BCS GENERATION COMMAND FILE: | RTPC | N/A | RTPC | N/A | N/A | N/A |
| FXL COMMAND FILE: | RTCM | MPCM1 | RTCM | MPCM5 | MPCM6 | MPCM7 |
| FXL TRANSFER FILE: | MPTRØ | MPTR1 | MPTR2 | MPTR5 | MPTR6 | MPTR7 |
| SYSTEM FILE FOR "COPYF": | DCPSYS | MPSYS | DCPSYS | MPSYS | MPSYS | MPSYS |
| OVERLAY FILE: | N/A | RAF | N/A | RAF | RAF | RAF |
| SPECIAL GENERATION PROCEDURES: | TR,MPTRØ<br>TR,RTTR | TR,MPTR1 | TR,DCPTR<br>TR,RTTR | TR,MPTR5 | TR,MPTR6 | TR,MPTR7 |

BCS SYSTEM GENERATION: MPSYS

FXL TRANSFER FILE: MPTR

FXL COMMAND FILE: BR1CSC

BCS COMMAND FILE: RAFPC

SPECIAL GENERATION PROCEDURE: TR,MPTR

DISC FORMATTING PROCEDURE:

RU,FMTXX1

TABLE 3.3.-1

## 4.0 REFERENCES

### 4.1 APPLICABLE DOCUMENTS

Following is a list of manufacturer's documents that describe the software and operator interface to the special purpose OLAAS PROCESSING & DISPLAY SYSTEM and the peripheral equipment. These documents contain supplementary information which is not duplicated in this manual.

HEWLETT-PACKARD COMPANY

| | |
|---|---|
| 92060-90012 | RTE-II: A Guide for New Users |
| 92001-93001 | REE-II Programming and Operating Manual |
| 92060-90014 | RTE Interactive Editor Reference Manual |
| 24998-90001 | DOS-RTE Relocatable Library Reference Manual |
| 5951-1321 | HP Fortran IV reference Manual |
| 92060-90005 | RTE Assembler Reference Manual |
| 92060-90010 | RTE & BSM Pocket Guide |

HP Fourier Analyzer System 5451B

| | |
|---|---|
| 05451-90199 | System Operating Manual |
| 05451-90268 | Supplement: March 1975 - System Operating Manual |
| 02-5952-0651 | Fourier Analyzer Training Manual (Application Note 140-0) |
| 05451-90333 | Magnetic Tape/Disc Option |
| 05451-90307 | Magnetic Tape/Disc Utility (MADIU) |
| 05451-90396 & 05451-90434 | Supplements: MADIU |
| 05451-98393 | Fourier Cross Loader FXL Operating & Software Note |

HP PERIPHERALS

| | |
|---|---|
| 12539-90008 | Time Base Generator |
| 12566-90015 | Microcircuit |
| 12908-90011 | Writeable Control Store |
| 05451-90217 | Data/Control Interface |
| 12880-90001 | Keyboard-Display Terminal |
| 12531-90033 | Buffer Teleprinter Interface |
| 02752-90004 | Teleprinter 2752A |
| 12597-90022 | Tape Reader |
| 02748-90032 | Tape Reader 2748B |

TEKTRONIX, INC.

| | |
|---|---|
| | Tektronix 4012 Computer Display Terminal User Instruction Manual |
| 062-1428-01 | Plot-10/MiniComputer - 4010 Series User Manual |

## 5.0 OLAAS SOFTWARE DOCUMENTATION (Programs/Subroutines Descriptions, Flow Diagrams and listings)

### 5.1 SOFTWARE OVERVIEW

The OLAAS Software System uses the Hewlett Packard 5451B FOURIER ANALYZER System hardware, including a HP7900 Disc Memory. Because of the size of the OLAAS System programs, and a need to maintain an initialized version of the analysis and data collection programs, it was necessary to segment the programs into separate coreloads. The HP2100S Computer has 32K of core memory, and the total system can accommodate 8 Fourier coreloads, using the HP7900 Disc Memory. The coreloads are organized as shown:

| CORELOAD | FUNCTION |
|---|---|
| 0 | OLAAS Execution Program |
| 1 | OLAAS Analysis and System Executive Subroutine |
| 2 | OLAAS Raw Data Collection Subroutine |
| 3 | Copy of coreload 1 (used for processing) |
| 4 | Copy of coreload 2 (used for multi-ping recording) |
| 5 | OLAAS User Programs Analysis |
| 6 | Fourier Analyzer Graphics |
| 7 | OLAAS Plot Display Subroutine |

The key to the system is the transfering of control from one coreload to another (core swapping or overlay).

The core swapping uses the core dump (CDUMP) and core load (CLOAD) subroutines. The core dump (CDUMP) routine writes all of core into a specified coreload area of the disk. The core load (CLOAD) routine reads a specified coreload and transfers to an address given in the calling sequence. Because the coreload routine must be overlaid by the program it is reading, it uses NIBBL, the H.P. loader which resides in an area of core (77600-77677) which is not overlaid. For this reason, NIBBL must be configured into the system. CLOAD modifies NIBBL so that when the coreload is completed, it transfers to the address specified in the CLOAD calling sequence. For the data collection coreload, it is 2, and for any coreload containing the Fourier software, it is the address of USTRT, which must be the first program unit loaded after the BASIC System and must be in base page. (This address is usually 51.) USTRT transfers control to Label 0 (zero) in the keyboard program stack. The way that the analysis program, or any other program in any other coreload, is started immediately, rather than requiring operator action.

The OLAAS Software System is composed of four (4) major functions which direct the processing of data. These functions are executive control, analysis and system control, raw data collection, and detection plot display. Additional function, which are part of system, user program and graphic displays are called separately by the executive.

The software was written in a modular approach which serves to minimize the amount of core memory storage required for the whole system. Therefore, the major functions required various subroutines to accomplish the overall task. The major function with the necessary subroutines is used to make the various coreloads. This section contains all the programs and subroutines used by the system. Also included are the user programs, and the TRANSFER, COMMAND, and PCS ANSWER Files used to generate the system and/or various coreload.

### 5.1.1. BASIC OLAAS SYSTEM CONTROL LOGIC

The Basic OLAAS System Control Logic is shown in Figure 5.1-1. The following description is in support of that figure. For detail operations refer to the flow diagram for each program or subroutine. Note that control is based on the use of "Core Swapping" and the "Not-First-Time Switch" (NFTSW).

The OLAAS System is loaded and enters the EXECUTIVE program, Y0003, which displays the system operation message. The operator selects the desired operation by input of the correct letter. Control is transfered to the proper coreload. The available operation are: D (Data Recording and Analysis), F (User Program Analysis), and G (Fourier Analyzer Graphics).

(1) The operator selects a "D" and control is transfered to coreload 1 (Y1002). Initially the NFTSW=1 and the system is initialized. NFTSW is set to "5" and control is transfered to coreload 3 (Y1002), which is a copy of coreload 1, for processing. NFTSW=5 causes the operating mode questions to be asked.

(2) The operator selects "3" and the date may be changed or entered and control returns to ask the operating mode questions again.

(3) The operator selects "1" and the On-Line Analysis questions are asked NFTSW is set to "2" and control is transfered to coreload 2 (Y0001) for Raw Data Collection.

(4) The first time Y0001 is entered the recording questions are aksed and then raw data is collected. A copy of coreload 2 is written into coreload 4 for multi-ping recording, and control is transfered to coreload 3 (Y1002).

(5) NFTSW=2 and the single ping analysis is done for the ping just recorded. NFTSW is set to "4" and control is transfered to coreload 7 (Y1007) for an analysis display.

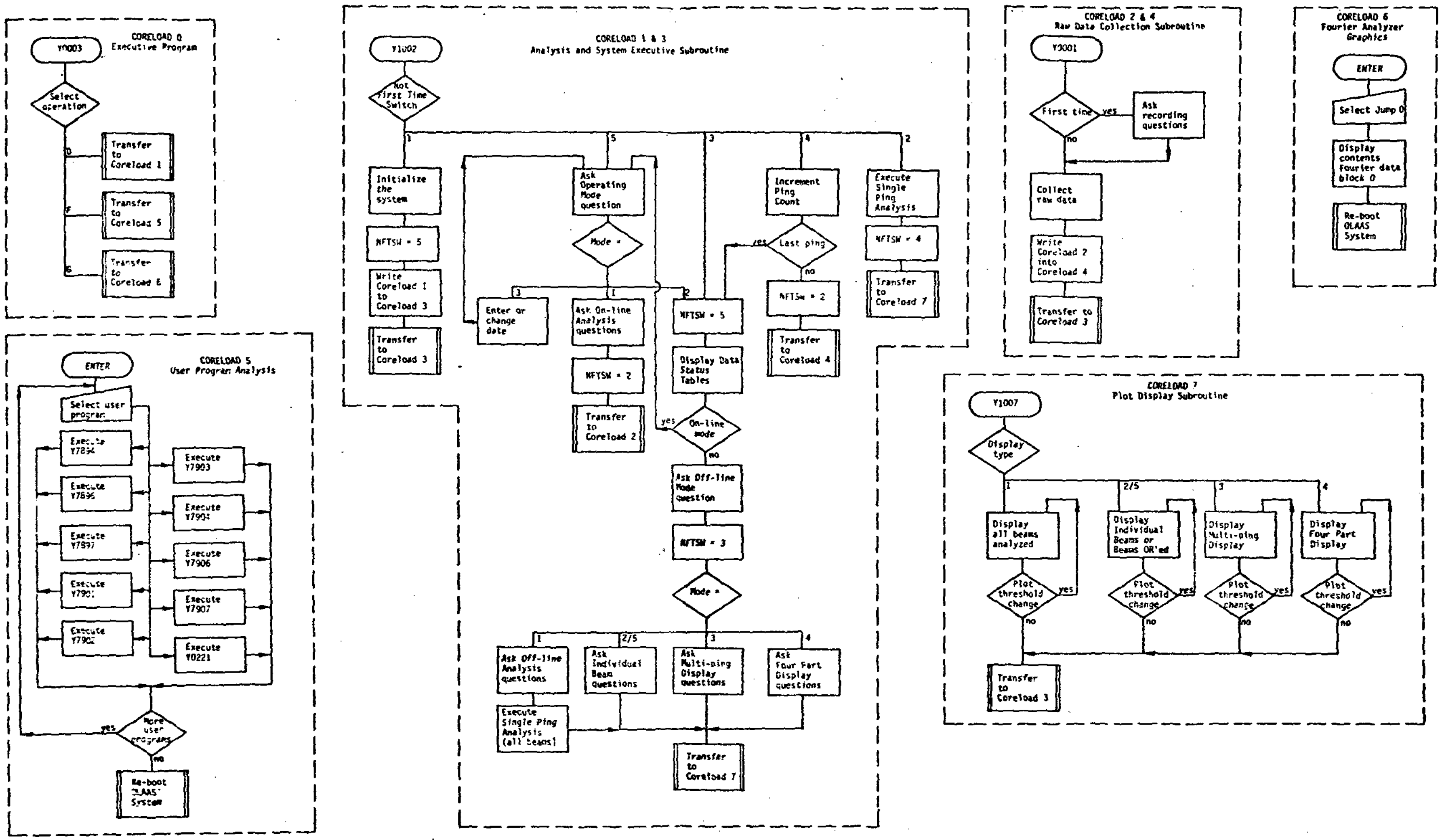



Figure 5.1-1. Basic OLAAS System Control Logic

(6) Test for the type of display is made and the plot is displayed. A check is made for a plot threshold change and if none control is transfered to coreload 3 (Y1002).

(7) NFTSW=4 and the ping count is incremented and checked for "last ping". If not last ping, the NFTSW is set to "2" and control is transfered to coreload 4 (Y0001) to record the next ping.

(8) This is not the first time Y0001 was entered, therefore the same parameter are used to collect raw data for the next ping. A new copy of coreload 2 is written into coreload 4 and control is transfered to coreload 3 (Y1002).

(9) NFTSW=2 repeats operation (5) and (6) above.

(10) NFTSW=4 and the ping count is incremented and checked for "last ping". If last ping, then NFTSW is set to "5" and the DATA STATUS TABLES are displayed. A check for ON-LINE mode is made and being set, control returns to ask the operating mode questions again.

(11) The operator selects "2" and the NFTSW is set to "5" and the DATA STATUS TABLES are displayed. A check for ON-LINE mode is made and being off, the OFF-LINE mode question is asked. The operator selects the desired off-line mode by entering the correct value. The available mode are: 1 (Analysis of Previously Recorded Data), 2 (Individual Beam Display), 3 (Multi-Ping Display), and 4 (Four Part Display).

(12) Note that the control in the "Off-Line" mode is basic the same for each selection. Therefore, this description is only for Analysis of Previously Recorded Data selection and the other selections shall reference this flow.

(13) The operator selects "1" and the NFTSW is set to "3". The Off-LIne Analysis questions are asked and Single Ping Analysis is executed, for all beam requested. Control is transfered to coreload 7 (Y1007) for the analysis display.

(14) Test for the type of display is made and the plot is displayed. A check is made for a plot threshold change and if none, control is transferred to coreload 3 (Y1002).

(15) NFTSW=3 and the NFTSW is set to "5". The DATA STATUS TABLES are displayed and the ON-LINE mode is checked. Being Off-Line mode question is asked.

(16) With analyzed data available the operator may select any of the other three (3) off-line modes. The display questions will be aksed and control is transfered to coreload 7 (Y1007) for the display. Operations (14) and (15) are repeated for each mode selection.

(17) The operator has two (2) options to restart the system. Enter a "12345" will cause control to return to coreload 0 (Y0003), Executive Program. Enter a "11111" will cause control to return to coreload 3 (Y1002) with NFTSW=5 and the operating mode question will be asked. Note that this option is only available when in coreload 3 (Y1003), Analysis and System Control Subroutines.

(18) The operator, when in the Executive Program (Y0003), may select User Program Analysis operation. Enter an "F" and control is passed to coreload 5, which contains the available user program.

(19) The operator enters the call for the user program selected and that program is executed. When execution is completed, the operator may select another user program or "re-boot" the OLAAS system to return control to the Executive Program (Y0003).

(20) The operator, when in the Executive Program (Y0003), may select Fourier Analyzer Graphics operation. Enter a "G" and control is passed to coreload 6, which contains the graphics package. Entering a "JUMP 0" command will cause the contents of Fourier Data Block 0 to be displayed. Re-boot the OLAAS system to return control to the Executive Program (Y0003).

### 5.1.2 SPECIAL SUBROUTINES

5.1.2.1 HEADR Subroutines. HEADR reads the header record from the disk. As there is a header record for each ping, the user specifies the ping number (first ping is ping 1) for which the header is to be read. Calling sequence is as follows:

```
CALL HEADR (IPING, IDATA)
```

where IPING is the ping number and IDATA is a 16 word array into which the header record is to be deposited. The content of the header record is as follows (all data items are integer format, with the exception of time, which is BCD):

| | | | |
|---|---|---|---|
| Word | 1 | TIME | time code, hours, BCD |
| | 2 | | time code, minutes and seconds, BCD |
| | 3 | ANGL1 | heading of beam 1, degrees |
| | 4 | | not used |
| | 5 | FSTB# | number of first beam recorded |
| | 6 | FBANG | heading of first beam recorded, degrees |
| | 7 | PING | ping number |
| | 8 | #BMS | number of beams recorded |
| | 9 | S | source (1=on line, 0=recorder) |
| | 10 | TOTPG | total number of pings (also known as M) |
| | 11 | K | delay before record, seconds |
| | 12 | N | length of record, seconds |
| | 13 | | reserved |
| | 14-16 | | not used |

5.1.2.2. SOUND Velocity Calculation Program. SOUND converts temperature verses depth profile to a sound velocity versus depth profile. A standard salinity of 34PPM is assumed at all depths. The following is the operating instructions for the SOUND program:

| STEP NO. | DESCRIPTION | OBSERVATION |
|---|---|---|
| 1 | Bootstrap load the RTE Software System and start the File Manager | RTE software system is loaded and file manager system is operating. |
| 2 | Enter and execute the following FMGR operator commands | |
| | :LG,1↓ | Allocate number tracks to load and go track area. |
| | :MR,SOUND↓ | Transfer relocatable program file to LG tracks. |
| | :RU,LOADR,99↓ | Run loader program, to load program from LG area |
| | :RU,SOUND↓ | Run SOUND program. |
| 3 | Enter a string of depths in feet and temperatures in degrees F separated by a comma. | Each entry is displayed. |
| 4 | Enter a temperature of 0 to terminate the string. | The SOUND velocity table is displayed. |
| 5 | Terminate operation by entering a negative number for depth. | Operation returns to FMGR. |

## 5.2 OLAAS PROCESSING AND DISPLAY SYSTEM

The OLAAS PROCESSING AND DISPLAY SYSTEM allows an operator to collect and analyze ERAPS Data subject to user-selected parameters. The user/operator may select the number of beams analyzed, look directions, range and doppler window. The type of display plotting is also under the user/operator control.

The following descriptions, flow diagrams, and listings are for the major functions of the system:

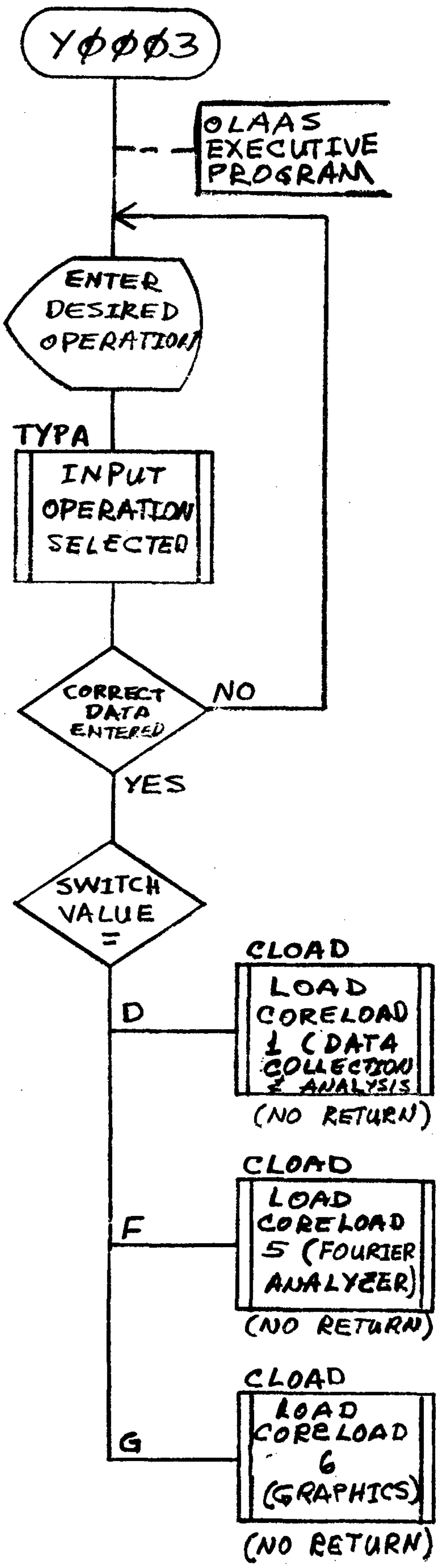
Y0003 - OLAAS Executive Program

Y1002 - OLAAS Analsis and System Executive Subroutine

Y0001 - OLAAS Raw Data Collection Subroutine

Y1007 - OLAAS Plot Display Subroutine

# OLAAS EXECUTIVE PROGRAM (YØØØ3) FLOW CHART



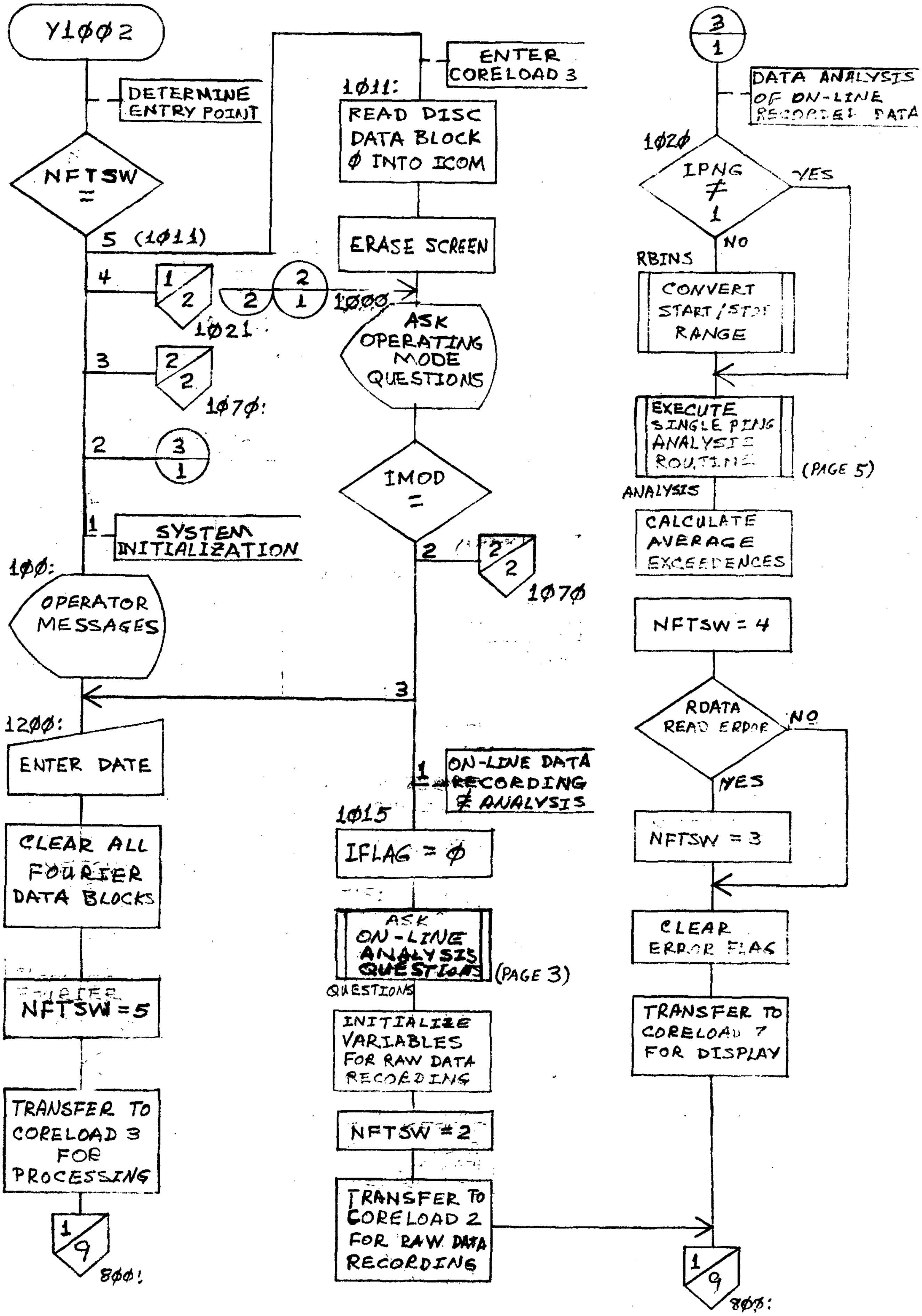
MP003A T=00004 IS ON CR00001 USING 00011 BLKS R=0020

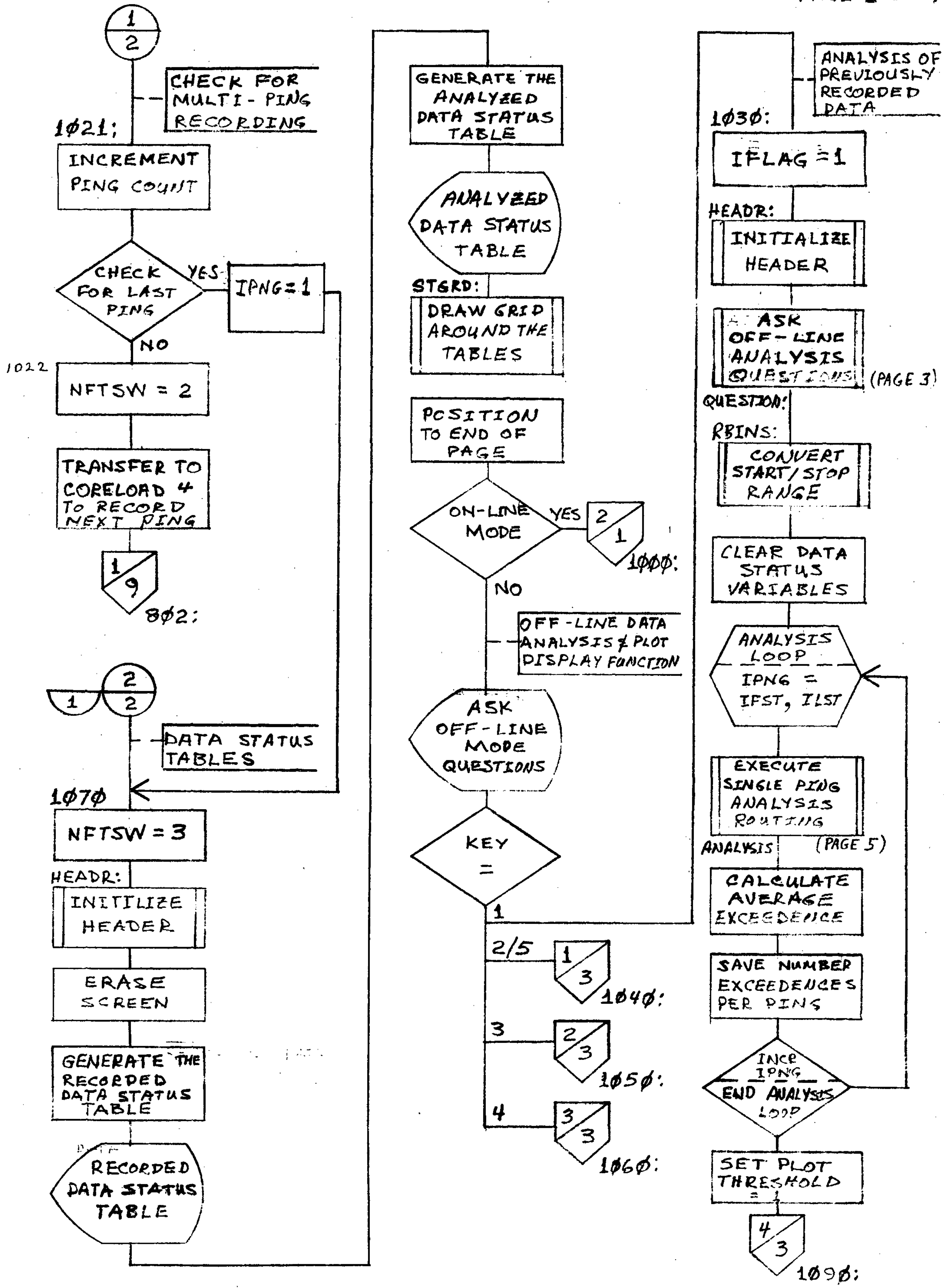
```
FTN
      PROGRAM Y0003
C   ************************************************************
C   *                                                          *
C   *  TITLE:   OLAAS EXECUTIVE PROGRAM                        *
C   *                                                          *
C   *  MODULE:  Y0003      SOURCE:  MP003A     VERSION:  X01   *
C   *                      OBJECT:  $Y0003                     *
C   *                                                          *
C   *  AUTHOR:  R. A. FULLERTON     DATE:  1976                *
C   *                                                          *
C   *  MODIFICATIONS:                                          *
C   *    1.  CHANGE THE EXECUTIVE QUESTION.                    *
C   *          12/09/77      W. BOWEN                          *
C   *                                                          *
C   *  MODULE FUNCTION:                                        *
C   *    THE OLAAS EXECUTIVE PROGRAM IS THE INITIAL ENTRY      *
C   *    INTO THE OLAAS PROCESSING & DISPLAY SYSTEM, AND IS    *
C   *    IN CORELOAD 0.   THE EXECUTIVE CALLS INTO EXECUTION   *
C   *    THE VARIOUS OPERATIONS OF THE SYSTEM (DATA COLLECTION,*
C   *    USER PROGRAM ANALYZER, & FOURIER ANALYZER GRAPHICS).  *
C   *    NOTE: THIS IS A BCS PROGRAM & NO FOURIER ANALYZER     *
C   *          SOFTWARE IS LOADED.                             *
C   *                                                          *
C   *  CALLING SEQUENCE:                                       *
C   *                                                          *
C   *     OLAAS SYSTEM BOOT LOAD PROCEDURE                     *
C   *                                                          *
C   ************************************************************


C > > > > DISPLAY THE EXECUTIVE QUESTIONS
```

```
      10 WRITE (2,20)
      20 FORMAT(//"*****  OLAAS PROCESSING & DISPLAY SYSTEM  *****"/
        +15X,"EXECUTIVE PROGRAM"//9X,
        +"** ENTER DESIRED OPERATION **"//
        +9X,"D = DATA RECORDING & ANALYSIS"/
        +9X,"F = USER PROGRAM ANALYZER"/
        +9X,"G = FOURIER ANALYZER GRAPHICS")
 C
         CALL TYPA (I,IFLG)

 C > > > > TRANSFER TO OPERATION SELECTED

         IF (IFLG) 10,10,30
 C
      30 I=I+20000B
         IF (I .EQ. 2H D) CALL CLOAD (07B,51B)
         IF (I .EQ. 2H F) CALL CLOAD (37B,2)
         IF (I .EQ. 2H G) CALL CLOAD (45B,2)
 C
         GO TO 10
 C
         END
         END$
```
:

Y1ØØ2 — ANALYSIS & SYSTEM EXECUTIVE SUBROUTINE

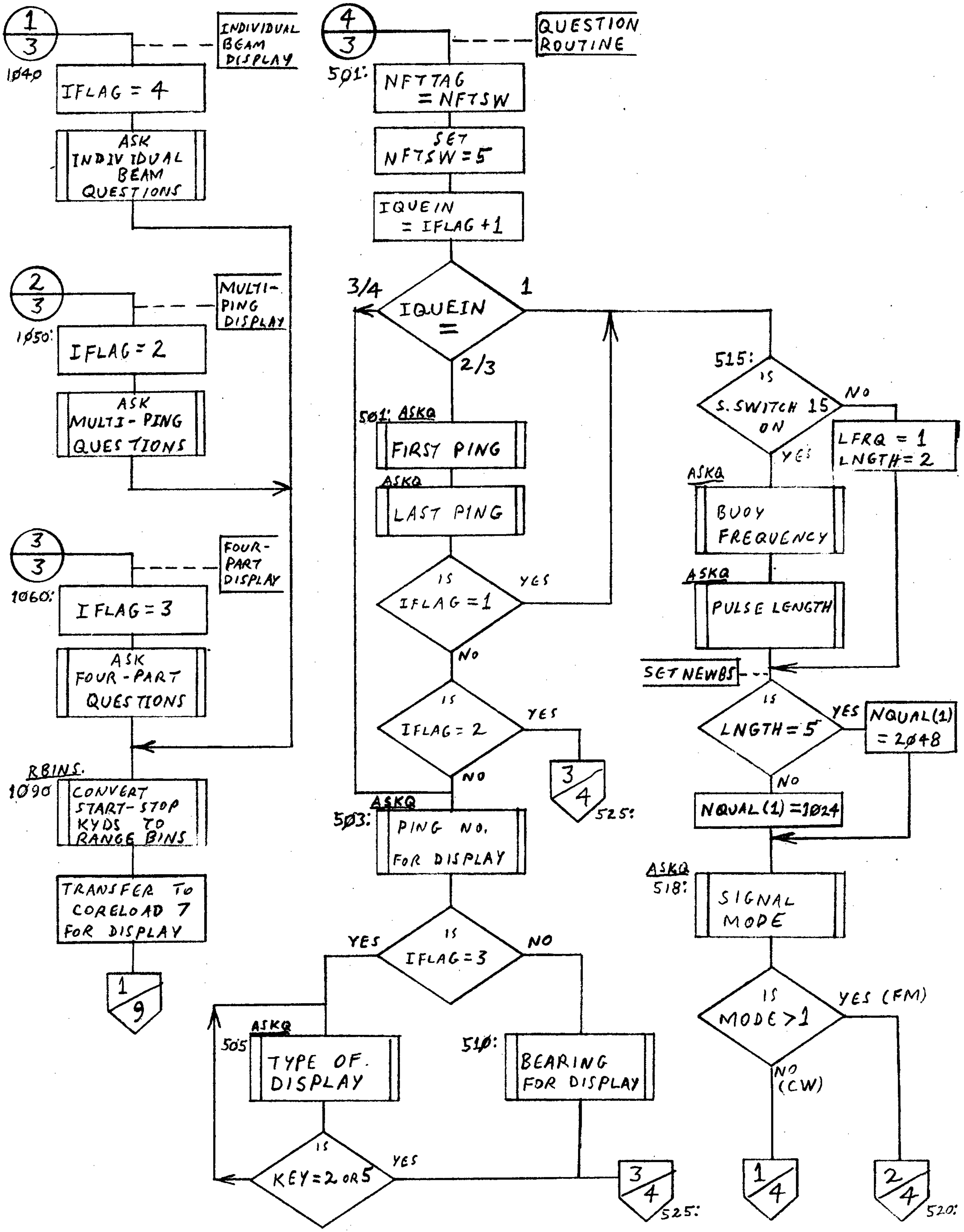


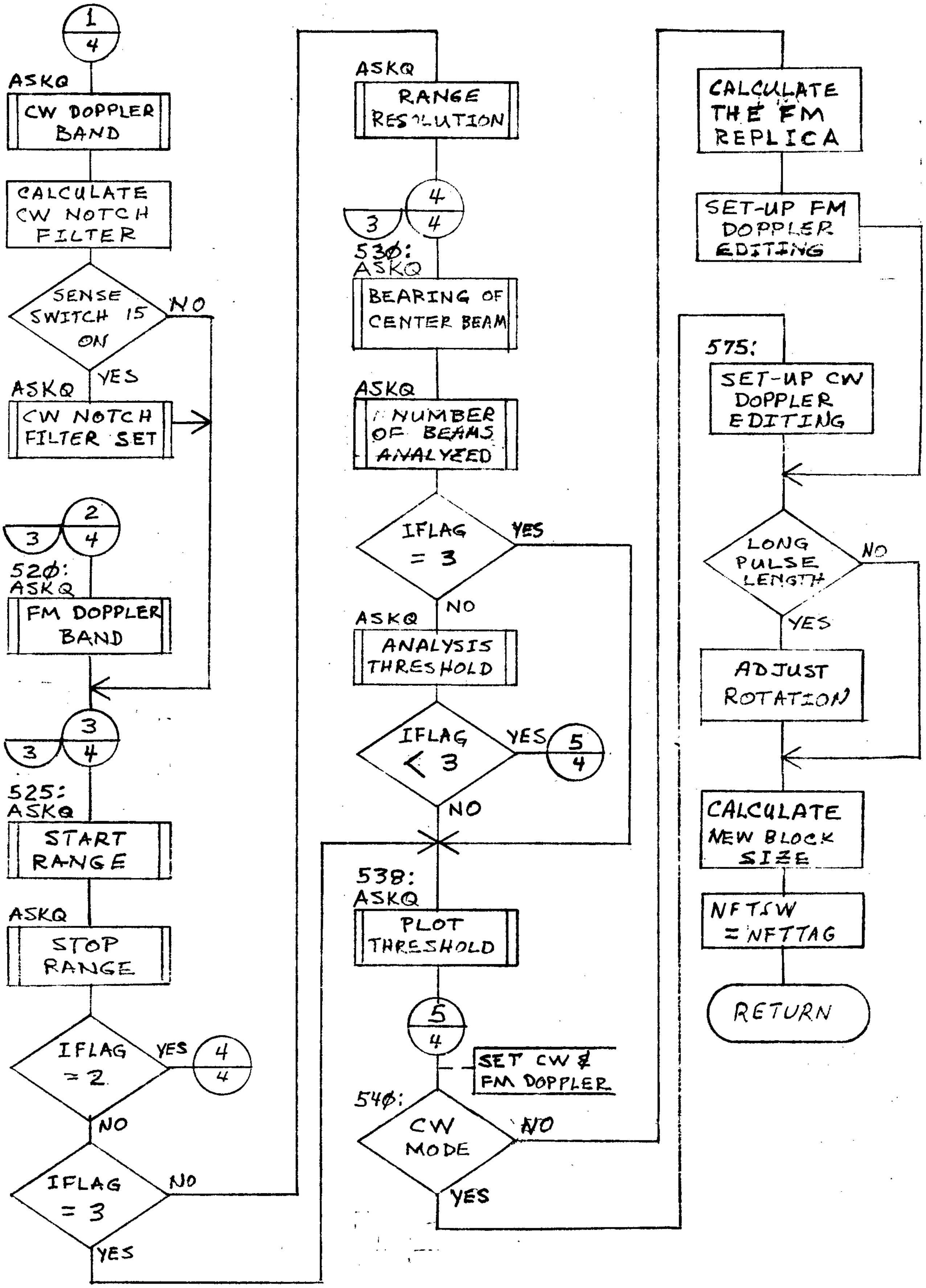


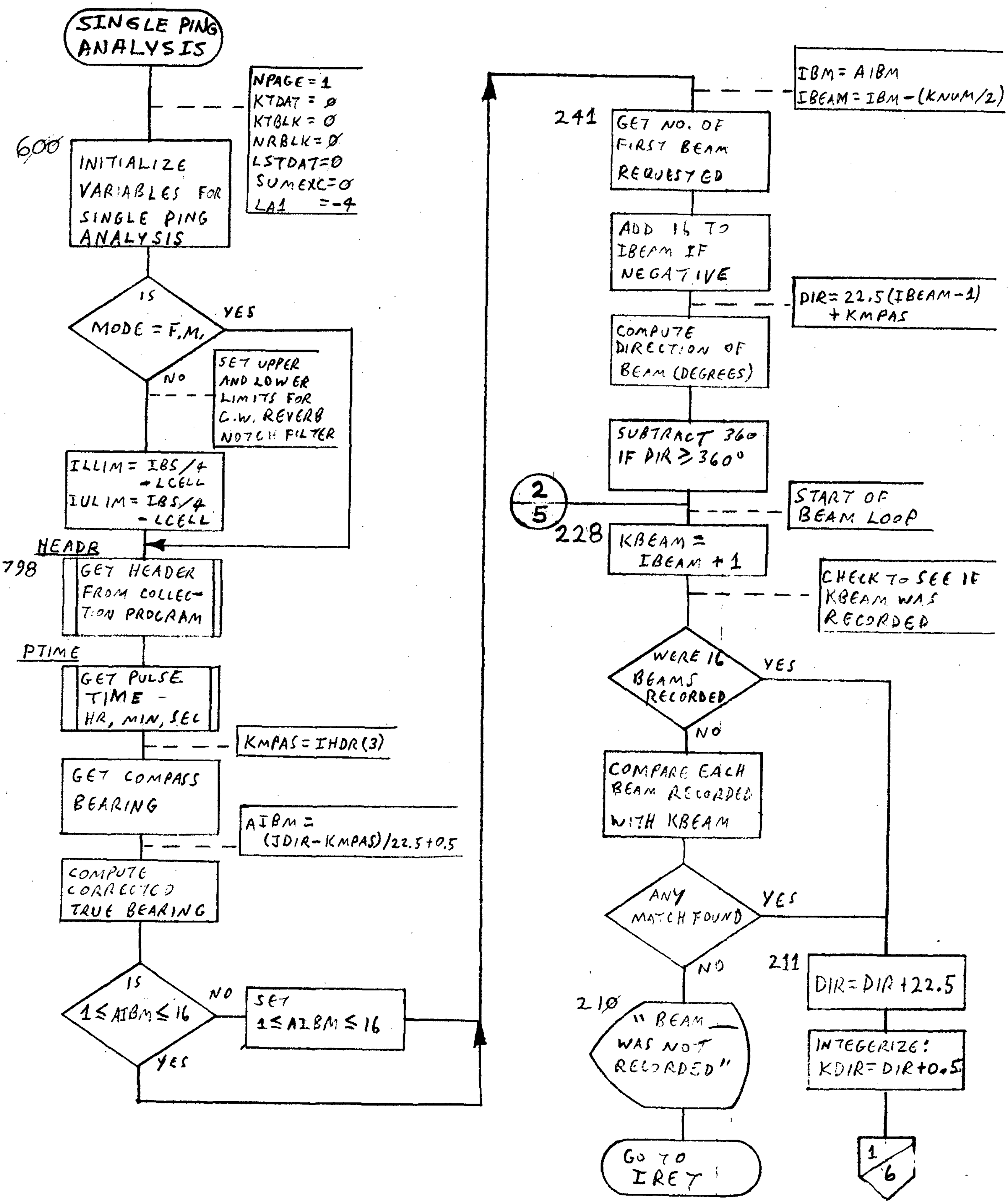

1/2
CHECK FOR MULTI-PING RECORDING
1Ø21:
INCREMENT PING COUNT
CHECK FOR LAST PING
YES
IPNG=1
NO
1022
NFTSW = 2
TRANSFER TO CORELOAD 4 TO RECORD NEXT PING
1/9
8Ø2:
2/2
1
DATA STATUS TABLES
1Ø7Ø
NFTSW = 3
HEADR:
INITILIZE HEADER
ERASE SCREEN
GENERATE THE RECORDED DATA STATUS TABLE
RECORDED DATA STATUS TABLE
GENERATE THE ANALYZED DATA STATUS TABLE
ANALYZED DATA STATUS TABLE
STGRD:
DRAW GRID AROUND THE TABLES
POSITION TO END OF PAGE
ON-LINE MODE
YES
2/1
1ØØØ:
NO
OFF-LINE DATA ANALYSIS & PLOT DISPLAY FUNCTION
ASK OFF-LINE MODE QUESTIONS
KEY =
1
2/5
1/3
1Ø4Ø:
3
2/3
1Ø5Ø:
4
3/3
1Ø6Ø:
ANALYSIS OF PREVIOUSLY RECORDED DATA
1Ø3Ø:
IFLAG = 1
HEADR:
INITIALIZE HEADER
ASK OFF-LINE ANALYSIS QUESTIONS
(PAGE 3)
QUESTION:
RBINS:
CONVERT START/STOP RANGE
CLEAR DATA STATUS VARIABLES
ANALYSIS LOOP
IPNG = IFST, ILST
EXECUTE SINGLE PING ANALYSIS ROUTING
ANALYSIS
(PAGE 5)
CALCULATE AVERAGE EXCEEDENCE
SAVE NUMBER EXCEEDENCES PER PING
INCR IPNG
END ANALYSIS LOOP
SET PLOT THRESHOLD
4/3
1Ø9Ø:

# Y1ØØ2 - ANALYSIS & SYSTEM EXECUTIVE SUBROUTINE



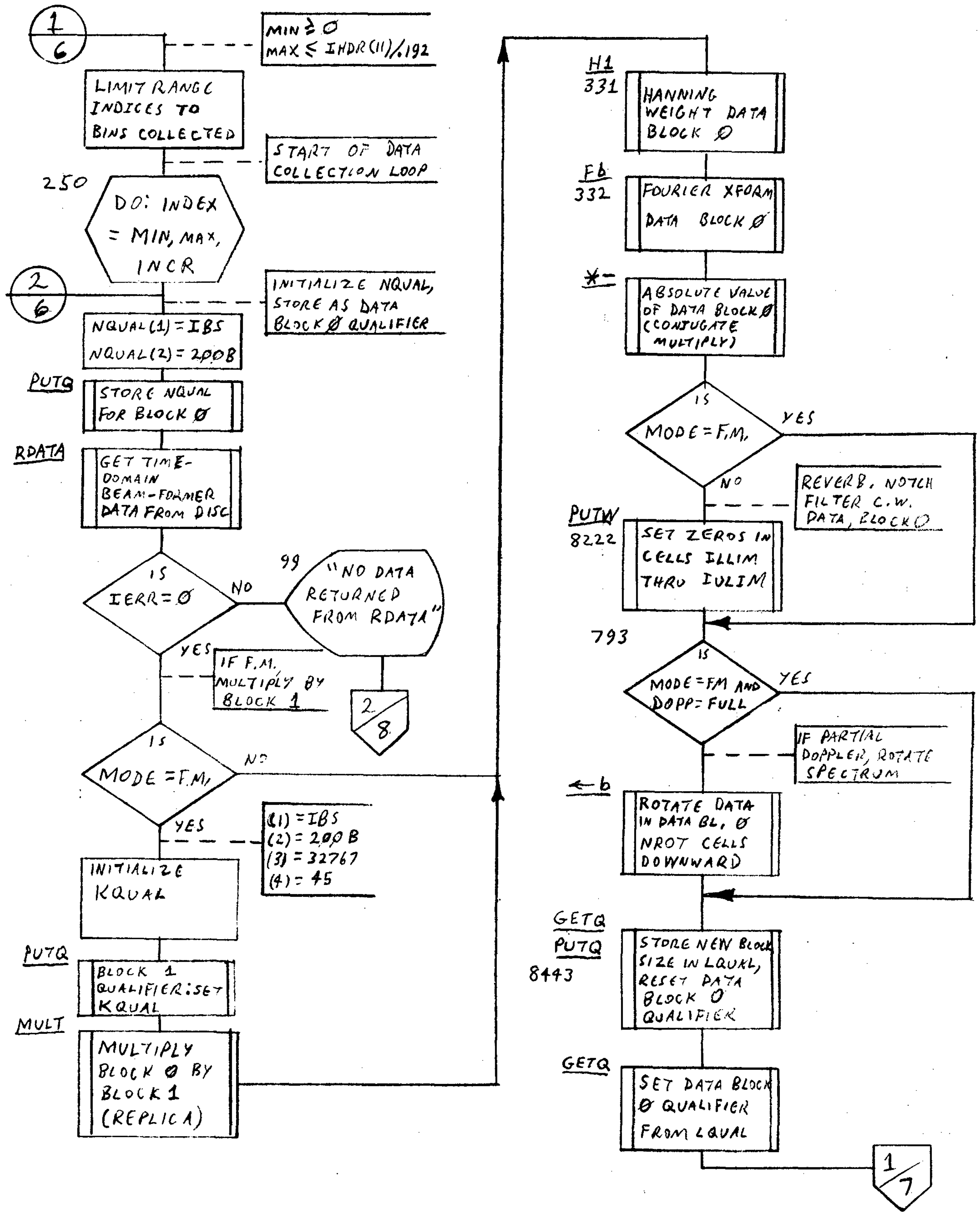



1/4
ASKQ
CW DOPPLER BAND
CALCULATE CW NOTCH FILTER
SENSE SWITCH 15 ON
NO
YES
ASKQ
CW NOTCH FILTER SET
3
2/4
52Ø:
ASKQ
FM DOPPLER BAND
3
3/4
525:
ASKQ
START RANGE
ASKQ
STOP RANGE
IFLAG = 2
YES
4/4
NO
IFLAG = 3
NO
YES
ASKQ
RANGE RESOLUTION
3
4/4
53Ø:
ASKQ
BEARING OF CENTER BEAM
ASKQ
NUMBER OF BEAMS ANALYZED
IFLAG = 3
YES
NO
ASKQ
ANALYSIS THRESHOLD
IFLAG < 3
YES
5/4
NO
538:
ASKQ
PLOT THRESHOLD
5/4
SET CW & FM DOPPLER
54Ø:
CW MODE
NO
YES
CALCULATE THE FM REPLICA
SET-UP FM DOPPLER EDITING
575:
SET-UP CW DOPPLER EDITING
LONG PULSE LENGTH
NO
YES
ADJUST ROTATION
CALCULATE NEW BLOCK SIZE
NFTSW = NFTTAG
RETURN


SINGLE PING ANALYSIS
NPAGE = 1
KTDAT = Ø
KTBLK = Ø
NRBLK = Ø
LSTDAT = Ø
SUMEXC = Ø
LA1 = -4
600
INITIALIZE VARIABLES FOR SINGLE PING ANALYSIS
IS MODE = F.M.
YES
NO
SET UPPER AND LOWER LIMITS FOR C.W. REVERB NOTCH FILTER
ILLIM = IBS/4 + LCELL
IULIM = IBS/4 - LCELL
HEADR
798
GET HEADER FROM COLLECTION PROGRAM
PTIME
GET PULSE TIME - HR, MIN, SEC
KMPAS = IHDR(3)
GET COMPASS BEARING
AIBM = (JDIR-KMPAS)/22.5+0.5
COMPUTE CORRECTED TRUE BEARING
IS 1≤AIBM≤16
NO
YES
SET 1≤AIBM≤16
IBM = AIBM
IBEAM = IBM-(KNUM/2)
241
GET NO. OF FIRST BEAM REQUESTED
ADD 16 TO IBEAM IF NEGATIVE
DIR = 22.5(IBEAM-1) + KMPAS
COMPUTE DIRECTION OF BEAM (DEGREES)
SUBTRACT 360 IF DIR ≥ 360°
2
5
START OF BEAM LOOP
228
KBEAM = IBEAM + 1
CHECK TO SEE IF KBEAM WAS RECORDED
WERE 16 BEAMS RECORDED
YES
NO
COMPARE EACH BEAM RECORDED WITH KBEAM
ANY MATCH FOUND
YES
NO
210
"BEAM ___ WAS NOT RECORDED"
GO TO IRET
211
DIR = DIR + 22.5
INTEGERIZE: KDIR = DIR + 0.5
1
6

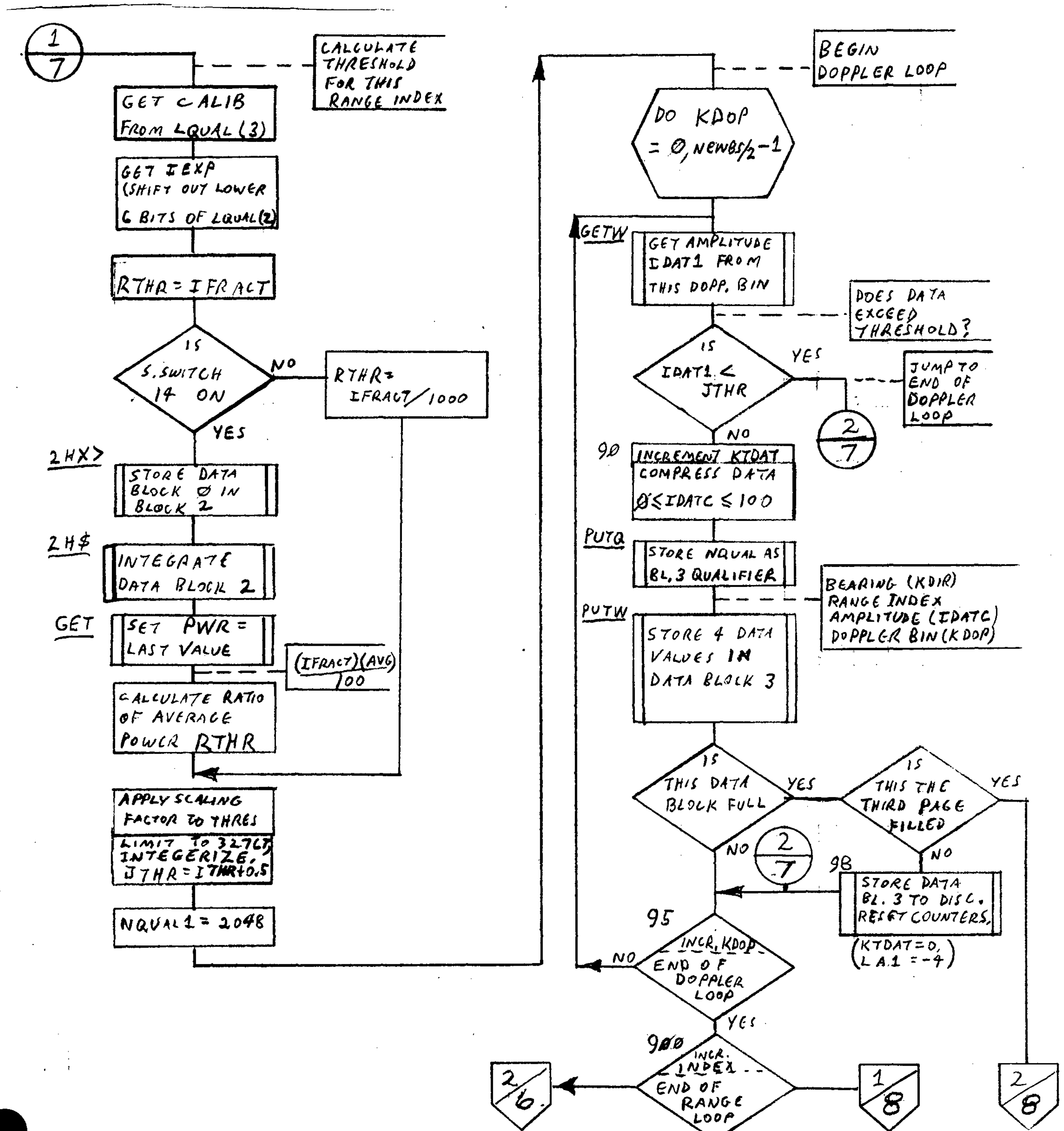
Y1ØØ2 - ANALYSIS & SYSTEM EXECUTIVE SUBROUTINE



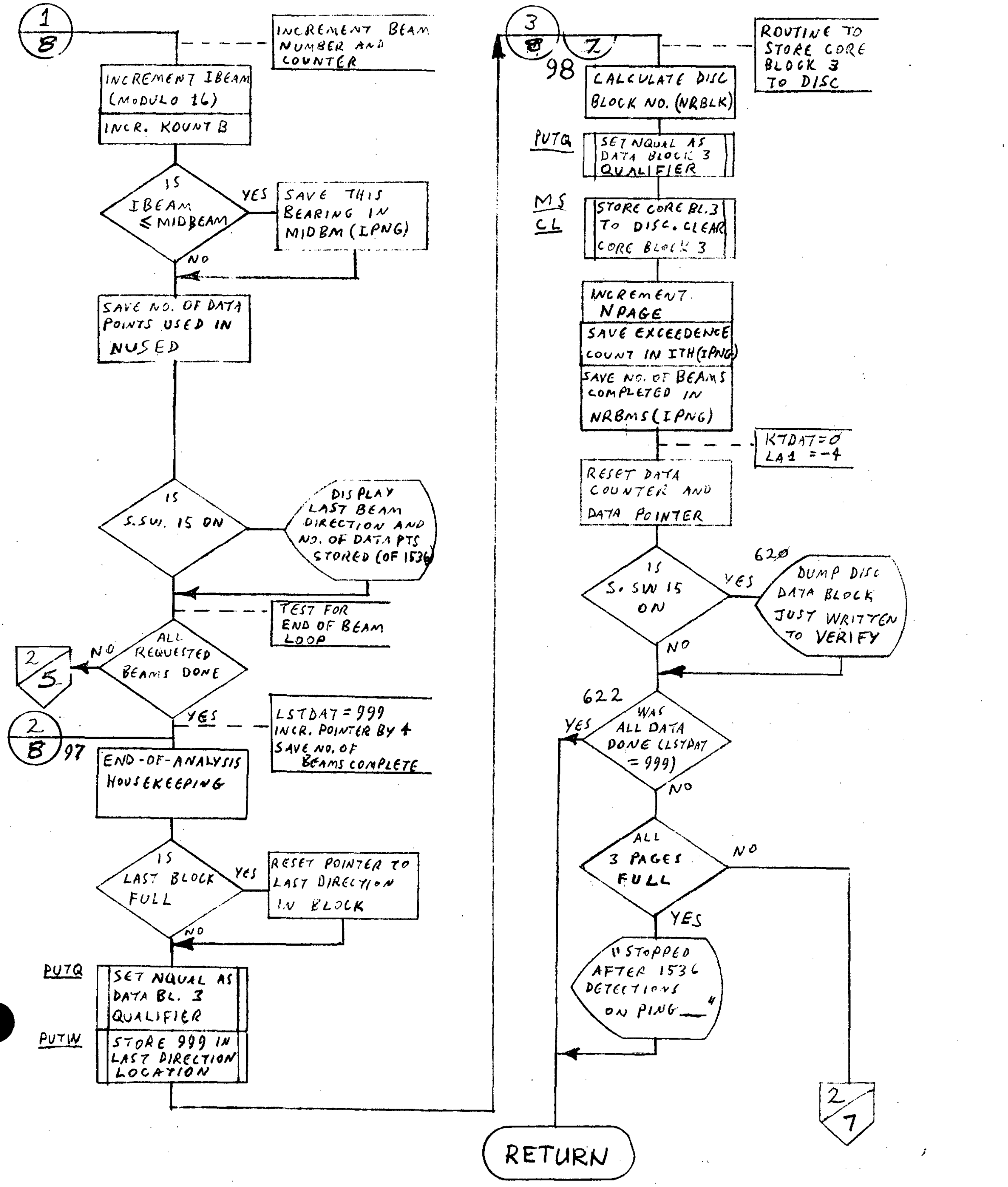


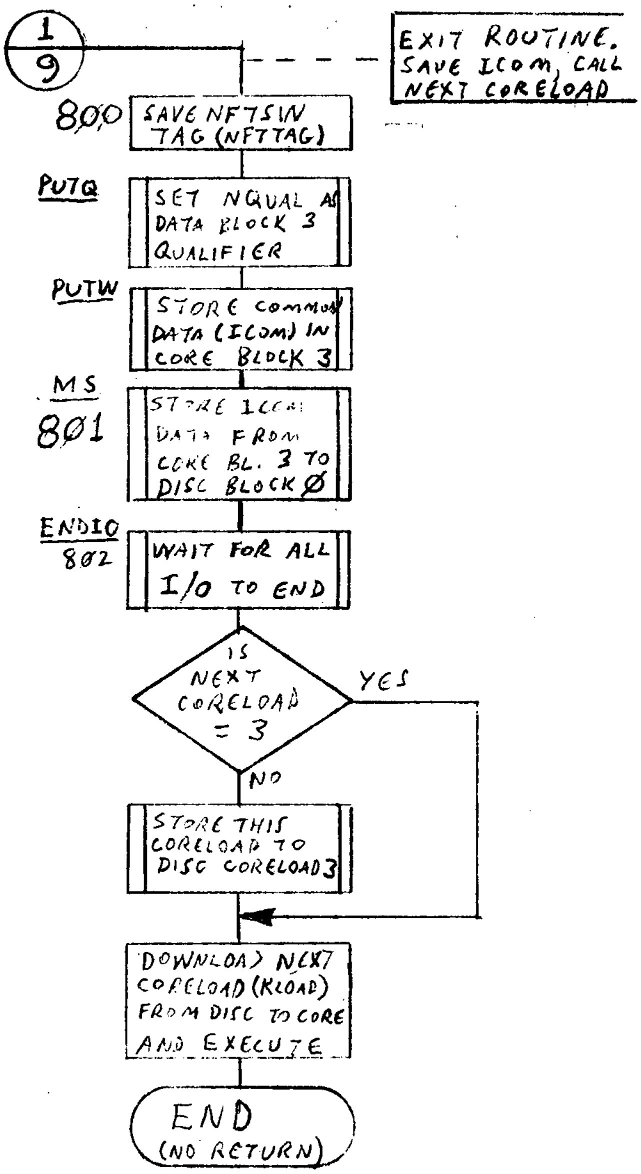
Y1ØØ2 - ANALYSIS & SYSTEM EXECUTIVE SUBROUTINE



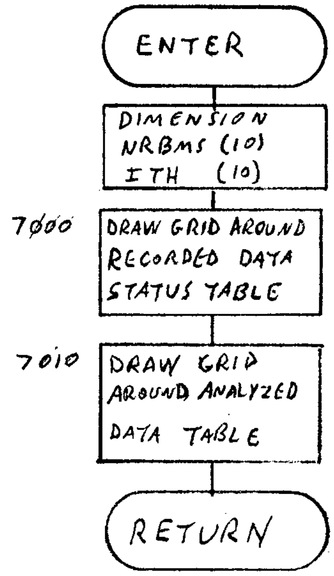


# Y1ØØ2 - ANALYSIS & SYSTEM EXECUTIVE SUBROUTINE

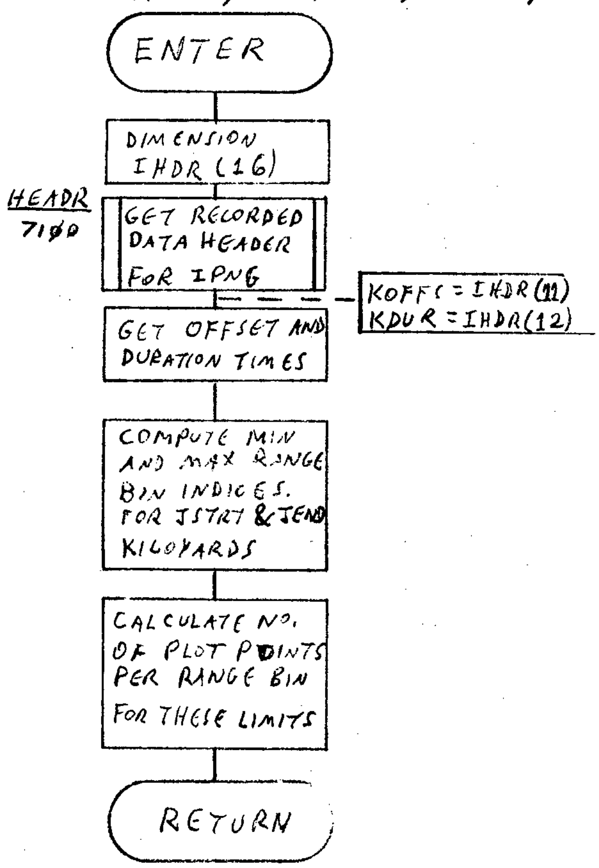




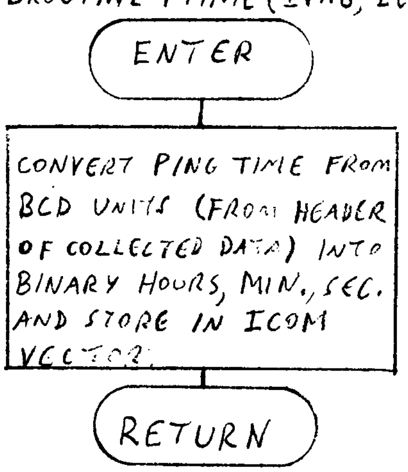

SUBROUTINE STGRD(NRCDS, NRBMS, ITH)




SUBROUTINE RBINS(IPNG, JSTRT, JEND, KOFFS, MIN, MAX, INCRX)




SUBROUTINE PTIME(IPNG, ICOM)



MP002A T=00004 IS ON CR00001 USING 00166 BLKS R=1045

```
FTN4
      SUBROUTINE Y1002
C                                     LATEST MODIFICATION:  2/6/78 (WPB)
C   **************************************************************
C   *                                                            *
C   *  TITLE:   ANALYSIS AND SYSTEM EXECUTIVE SUBROUTINE         *
C   *                                                            *
C   *  MODULE:  Y1002       SOURCE:  MP002A      VERSION:  X04   *
C   *                       OBJECT:  Y10020                      *
C   *                                                            *
C   *  AUTHOR:  D. L. GOODMAN        DATE:  1976                 *
C   *                                                            *
C   *  MODIFICATIONS:                                            *
C   *     1:  CHANGES FOR OLAAS SYSTEM PROBLEMS                  *
C   *           09/30/77 - B. TASHJIAN & W. BOWEN                *
C   *     2:  WRITES MULTI-PING DATA INTO 30 DATA BLOCKS.        *
C   *         DISC BLOCK 0 IS USED AS COMMON AREA, STORED        *
C   *         IN ICOM(128) IN FORTRAN.                           *
C   *           11/18/77 - C.R. MOOSE                            *
C   *     3:  CHANGE USER-ENTERED QUESTIONS                      *
C   *           11/30/77 - W. BOWEN                              *
C   *     4:  CHANGE TO CALCULATE THE ANALYSIS THRESHOLD         *
C   *             1/24/78 - W. BOWEN                             *
C   *                                                            *
C   *  MODULE FUNCTION:                                          *
C   *     Y1002 IS THE EXECUTIVE SUBROUTINE FOR THE OLAAS        *
C   *  ANALYSIS AND PERFORMS THE FOLLOWING FUNCTIONS:            *
C   *  1. COLLECTS USER-ENTERED PARAMETERS FOR THE ANALYSIS.     *
C   *  2. ON-LINE DATA RECORDING AND ANALYSIS.                   *
C   *  3. OFF-LINE DATA ANALYSIS.                                *
C   *  4. SINGLE PING AND BEAMFORMER DATA ANALYSIS.              *
C   *  5. MULTI-PING ANALYSIS.                                   *
```

```
C  *                                                                    *
C  *  CALLING SEQUENCE:     CALLED BY - Y0003 (OLAAS EXECUTIVE)         *
C  *                                                                    *
C  *     CALL CLOAD (07B,51B)                                           *
C  *                                                                    *
C  **********************************************************************

C  **********************************************************************
C  *  COMMON AREA ASSIGNMENTS  (ICOM AND/OR DISC DATA BLOCK 0)          *
C  **********************************************************************
C  *                                                                    *
C  *  (0)   JDIV: DIVISOR OF ORIGINAL BLOCK FOR DOPPLER EDITING         *
C  *  (1)   IBND: SPECIFIC DOPPLER ANALYSIS BAND (1-JDIV BANDS)         *
C  *  (2)  NEWBS: BLOCK SIZE OF DOPPLER BAND                            *
C  *  (3)    MIN: MINIMUM INDEX USED IN ANALYSIS BY RDATA PGM.          *
C  *  (4)    MAX: MAXIMUM INDEX USED IN ANALYSIS BY RDATA PGM.          *
C  *  (5)  IFRAC: ANALYSIS THRESHOLD PERCENTAGE SET BY USER             *
C  *  (6)   INCR: TIME INCREMENT IN MULTIPLES OF 0.192 SECONDS          *
C  *  (7)   KODE: HANNING EQUALS 1, NO HANNING EQUALS 0                 *
C  *  (8)  KTYPE: RANGE, KILOYARDS EQUALS 1, TIME EQUALS 0              *
C  *  (9)   IPNG: PING NUMBER                                           *
C  * (10)   KNUM: NUMBER OF BEAMS USED IN ANALYSIS                      *
C  * (11)  KOFFS: TIME OFFSET VALUE                                     *
C  * (12)   MODE: 0 = CW PULSE,  1 = FM UP,  2 = FM DOWN                *
C  * (13)  LNGTH: EITHER 2 OR 5 SECONDS                                 *
C  * (14)    KT1: TIME, HOURS                                           *
C  * (15)    KT2: TIME, MINUTES                                         *
C  * (16)    KT3: TIME, SECONDS                                         *
C  * (17)   LFRQ: BUOY FREQUENCY, 1 = F1, 2 = F2                        *
C  * (18)   JDIV: DIVISOR OF ORIG. BLOCK (SAME AS WORD 0)               *
C  * (19)  NTATE: DOPPLER AXIS ROTATION, 1 = YES, 0 = NO                *
C  * (20)  NSHFT: FOR NTATE=1, NO. OF KNOTS TO SHIFT DOPPLER            *
C  * (21)             SPARE                                             *
C  * (22)    KEY: 0 = ON-LINE PLOT                                      *
C  *              1 = ANALYSIS OF PREVIOUSLY RECORDED DATA              *
```

```
C   *                     2 = INDIVIDUAL BEAM DISPLAY                          *
C   *                     3 = MULTI-PING DISPLAY                               *
C   *                     4 = FOUR-PART DISPLAY                                *
C   * (23-32) ITH1: NUMBER OF WORDS STORED FOR EACH PING                       *
C   * (33)                SPARE                                                *
C   * (34) MXPNGS: MAXIMUM NUMBER OF PINGS TO BE COLLECTED                     *
C   * (35)    IFST: FIRST PING NO. FOR ANALYSIS                                *
C   * (36)    ILST: LAST PING NO. FOR ANALYSIS                                 *
C   * (37)  IDPLR: DOPPLER BAND (1=FULL, 2=HALF, 3=UP, 4=DOWN)                 *
C   * (38)  JSTRT: START RANGE (IN KYDS.)                                      *
C   * (39)    JEND: END   RANGE (IN KYDS.)                                     *
C   * (40)    JDIR: DIRECTION OF BEARING OF CENTER BEAM OR                     *
C   *               BEAN NUMBER TO DISPLAY FOR FOUR PART DISPLAY               *
C   * (41)    IMOD: MODE (ON-LINE / OFF-LINE ANALYSIS)                         *
C   * (42-46)             SPARES                                               *
C   * (47)  INCRX: LENGTH OF ONE RANGE INCREMENT (IN PLOT POINTS)              *
C   * (48)    KTHR: CURRENT PERCENTAGE FOR PLOT THRESHOLD (1-100)              *
C   * (49-50)             SPARE                                                *
C   * (51-66) IHDR: HEADER FROM INPUT DATA TAPE                                *
C   * (67-69)             SPARES                                               *
C   * (70) NFTTAG: DUPLICATE OF NOT-FIRST-TIME SWITCH                          *
C   * (71-80)  IAVGEX: NUMBER OF EXCEDENCES FOR ALL 10 PINGS                   *
C   * (81-90)   MIDBM: CENTER BEAM ANALYZED FOR ALL 10 PINGS                   *
C   * (91-100)  NRBMS: NUMBER OF BEAMS ANALYZED FOR ALL 10 PINGS               *
C   * (101-109) IDATE: ALPHANUMERIC FIELD FOR DATE (18 CHAR.)                  *
C   ************************************************************************


C   ************************************************************************
C   *  DEFINE ARRAYS, CONSTANTS, VARIABLES, & DATA                         *
C   ************************************************************************

      DIMENSION ICOM(128),NQUAL(5),LQUAL(5),IHDR(16),MQUAL(5),
     +IOFF(10),IBCD(6),MASK(4),KQUAL(5),KLTRK(8),KLADR(8),ITH(10),
     +IAVGEX(10),MIDBM(10),NRBMS(10),IDATE(9),IBUFD(9)
```

```
      EQUIVALENCE
     +(ICOM(  1),  IBND),(ICOM(  2), NEWBS),(ICOM(  3),   MIN),
     +(ICOM(  4),   MAX),(ICOM(  5), IFRAC),(ICOM(  6),  INCR),
     +(ICOM(  7),  KODE),(ICOM(  8), KTYPE),(ICOM(  9),  IPNG),
     +(ICOM( 10),  KNUM),(ICOM( 11), KOFFS),(ICOM( 12),  MODE),
     +(ICOM( 13), LNGTH),(ICOM( 14),   KT1),(ICOM( 15),   KT2),
     +(ICOM( 16),   KT3),(ICOM( 17),  LFRQ),(ICOM( 18),  JDIV),
     +(ICOM( 19), NTATE),(ICOM( 20), NSHFT),
     +(ICOM( 22),   KEY),(ICOM( 23),   ITH),
     +(ICOM( 34),MXPNGS),(ICOM( 35),  IFST),(ICOM( 36),  ILST),
     +(ICOM( 37), IDPLR),(ICOM( 38), JSTRT),(ICOM( 39),  JEND),
     +(ICOM( 40),  JDIR),(ICOM( 41),  IMOD),
     +                   (ICOM( 47), INCRX),(ICOM( 48),  KTHR),
     +                                      (ICOM( 51),  IHDR),
     +(ICOM( 70),NFTTAG),(ICOM( 71),IAUGEX),(ICOM( 81), MIDBM),
     +(ICOM( 91), NRBMS),(ICOM(101), IDATE)

      DATA NFTSW/1/,II/0/,IMOD/2/,IFST/1/,ILST/1/,LNGTH/2/
      DATA MODE/1/,IDPLR/1/,IBND/2/,KODE/1/,KTYPE/0/,JSTRT/8/
      DATA JEND/24/,KMIN/10/,KMAX/20/,INCRT/1/,JDIR/0/,KNUM/3/
      DATA IFRAC/400/,LFRQ/2/,ISKIP/0/,LSWP/0/,NTATE/0/,KEY/1/
      DATA NSHFT/14/,MPNG1/1/,MPNGKT/1/,IDATE/9*2H  /,IPNG/1/
      DATA KLTRK/1B,7B,15B,23B,31B,37B,45B,53B/,IAUGEX/10*0/
      DATA KLADR/2B,51B,2B,51B,2B,2B,2B,51B/,MIDBM/10*999/
      DATA MQUAL/2048,100B,16383,45,0/,NRBMS/10*0/
      DATA NQUAL/2048,100B,16383,45,0/
      DATA ITH/10*0/,KTHR/1/

C   ****************************************************************
C   *  SUBROUTINE ENTRY POINT                                      *
C   ****************************************************************

C > > > > DETERMINE ENTRY POINTS
      GO TO (100, 1020, 1070, 1021, 1000) NFTSW
```

```
C   ****************************************************************
C   *  CORELOAD 1 ENTRY POINT  (NFTSW = 1)                         *
C   *     SUBROUTINE INITIALIZATION                                *
C   ****************************************************************

C > > > > DISPLAY OPERATOR'S MESSAGES
  100 WRITE  (2,105)
  105 FORMAT (//4X,"*****  DATA RECORDING & ANALYSIS  *****"
     +//13X,"SYSTEM INITIALIZATION"//"OPERATOR NOTE:  SET SENSE"
     +" SWITCH 15 (ON) TO CHANGE THE BUOY"/16X,"FREQUENCY, PULSE"
     +" LENGTH, & CW NOTCH FILTER.")

C > > > > ENTER DESIRED DATE
  110 DO 115 I=1,9
  115 IBUFD(I) = 2H
  120 WRITE  (2,125) IDATE
  125 FORMAT (/"ENTER DATE"/"(",9A2,")")
      CALL TYPC (IBUFD,IFLG,18)
      IF (IFLG) 120,140,130
  130 DO 135 I=1,9
  135 IDATE(I) = IBUFD(I)
  140 IMOD = 2
      IF (IMOD .EQ. 3) GO TO 145

C >> DIAGONSTICS >>  SENSE SW. 3 (ON) GO TO CORELOAD 3 DIRECTLY
        IF (ISSW(3) .LT. 0)  NFTSW = 3
        IF (ISSW(3) .LT. 0)  CALL CLOAD (23B,51B)

C > > > > DISABLE MAXIMIZATION ROUTINE
      CALL KYBD (2HY ,3010,1)

C > > > > INITIALIZE FOURIER DATA BLOCKS
      CALL KYBD (2HBS,2048)
      CALL KYBD (2HCL,0)
      CALL KYBD (2HCL,1)
```

```
      CALL KYBD (2HCL,2)
      CALL KYBD (2HCL,3)

C > > > > TRANSFER TO COREL0AD 3 FOR PROCESSING
  145 NFTSW = 5
      KLOAD = 3
        GO TO 800

C   #####################  LEAVE CORELOAD 1  ######################


C   ***************************************************************
C   *  CORELOAD 3 ENTRY POINT   (NFTSW = 5)                       *
C   *     SUBROUTINE PROCESSING                                   *
C   ***************************************************************

C > > > > READ IN DISC DATA BLOCK 0 INTO ICOM
 1000 CALL PUTQ (3,NQUAL)
      CALL KYBD (2HMS,31,0)
      CALL KYBD (2HMS,11,3)
      DO 1005 J = 1, 128
        CALL GETW (3, J, ICOM(J), II)
 1005 CONTINUE

C >> DIAGONSTICS >>  SENSE SW. 10 (ON) WRITES ICOM AREA
        IF (ISSW(10) .LT. 0) WRITE (1,9999) ICOM
        IF (ISSW(10) .LT. 0) PAUSE 201

C > > > > ERASE SCREEN
      CALL CHOUT(33B)
      CALL CHOUT(14B)
      CALL D750

C > > > > ASK OPERATING MODE QUESTIONS
 1010 CALL ASKQ (IMOD,1,3,19,IDATE)
```

```
      IF (IMOD .EQ. 1) GO TO 1015
      IF (IMOD .EQ. 2) GO TO 1070
      IF (IMOD .EQ. 3) GO TO 110

C   ****************************************************************
C   *  ON-LINE DATA RECORDING AND ANALYSIS FUNCTION                *
C   ****************************************************************

C > > > > ASK ON-LINE ANALYSIS QUESTIONS
 1015 IFLAG = 0
      WRITE(1,1300)
 1300 FORMAT(/"*****  ON-LINE DATA RECORDING & ANALYSIS"
     +"  *****"//5X,"**  ANSWER ANALYSIS QUESTIONS  **")
      ASSIGN 1100 TO IRET
        GO TO 500

C > > > > RE-ENTER FROM QUESTION SUBROUTINE
 1100 CONTINUE

C > > > > INITIALIZE VARIABLES FOR RAW DATA RECORDING
      IPNG = 1
      IFST = 1
      DO 1302 I = 1,10
1302  ITH(I) = 0
      IOFF(1) = 30

C > > > > TRANSFER TO CORELOAD 2 FOR RAW DATA RECORDING
      NFTSW = 2
      KLOAD = 2
        GO TO 800

C   *********************  LEAVE CORELOAD 3  *********************
```

```
C   ****************************************************************
C   *  ENTRY POINT ON RETURN FROM CORELOAD 2  (NFTSW = 2)            *
C   *     ON-LINE DATA ANALYSIS & DISPLAY                            *
C   ****************************************************************

1020  CONTINUE
      IF (IPNG .NE. 1)  GO TO 1102

C > > > > CONVERT START/STOP RANGE
      CALL RBINS(IPNG, JSTRT, JEND, KOFFS, MIN, MAX, INCRX)
 1102 MQUAL(1) = 2048

C > > > > TRANSFER TO SINGLE PING ANALYSIS
      ASSIGN 1105 TO IRET
        GO TO 600

C > > > > RE-ENTER FROM SINGLE PING ANALYSIS SUBROUTINE
1105  CONTINUE
      AVGEXC = SUMEXC / ITH(IPNG)
      IAVGEX(IPNG) = AVGEXC
      IF (IPNG .EQ. 1) WRITE(1, 9998)
 9998 FORMAT (" ")

C >> DIAGONSTICS >>  SENSE SW. 10 (ON)
C     IF (ISSW(10) .LT. 0) WRITE(1, 9105)IPNG,ITH(IPNG),AVGEXC
C9105 FORMAT(" PING NO.",I3," HAS",I5," EXCEEDENCES.",
C    + "  THE AVERAGE VALUE IS",F10.1)

C > > > > TRANSFER TO COREL0AD 7 FOR PLOT
      KTHR = 1
      NFTSW = 4
C \/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/
C > > > > RECOVERY FROM BAD RDATA READ ERROR
          IF(IERR .NE. 0)  NFTSW = 3
          IERR = 0
```

```
C \/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/
      KLOAD = 7
        GO TO 800

C   #####################  LEAVE CORELOAD 3  #####################


C   *************************************************************
C   *  ENTRY POINT ON RETURN FROM CORELOAD 4  (NFTSW =4)        *
C   *     ON-LINE MULTI-PING RECORDING                          *
C   *************************************************************


1021  IPNG = IPNG + 1

C > > > > CHECK TO SEE IF LAST PING
      CALL HEADR(1,IHDR)

C >> DIAGONSTICS >>  SENSE SW. 10 (ON)
C     IF(ISSW(10) .LT. 0) WRITE(1, 9102) (ICOM(J), J = 51, 66)
C9102   FORMAT(" HDR ", 11I6)

      IF(IPNG .LE. IHDR(10))  GO TO 1022
      IPNG = 1
        GO TO 1070

C > > > > TRANSFER TO CORELOAD 4 TO RECORD THE NEXT PING
1022  NFTSW = 2
      KLOAD = 4
        GO TO 802

C   #####################  LEAVE CORELOAD 3  #####################
```

```
C   ****************************************************************
C   *  ENTRY POINT TO DISPLAY THE DATA STATUS TABLES               *
C   *       (1) FOR OFF-LINE ANALYSIS & PLOT DISPLAY               *
C   *       (2) ON RETURN FROM CORELOAD 7  (NFTSW = 3)             *
C   *                OFF-LINE ANALYSIS & PLOT DISPLAY              *
C   *       (3) TO DISPLAY ON-LINE DATA STATUS  (NFTSW = 5)        *
C   ****************************************************************

C > > > > OLAAS DATA PARAMETER STATUS TABLES
 1070 NFTSW = 5
      CALL HEADR(1,IHDR)
      IEND = IHDR(11)+IHDR(12)

C > > > > ERASE SCREEN
      CALL CHOUT(33B)
      CALL CHOUT(14B)
      CALL D750

C > > > > RECORDED DATA STATUS TABLE GENERATION
      WRITE (2,1071) IDATE
 1071 FORMAT ("***********  OLAAS PROCESSING & DISPLAY "
     +"SYSTEM  ************"/21X,9A2/)

      WRITE (2,1072) IHDR(10),IHDR(11),IEND,IHDR(8)
 1072 FORMAT (16X,"RECORDED DATA STATUS TABLE"/7X,
     +"NUMBER OF PINGS RECORDED =",I3,/7X,
     +"PING RECORD WINDOW       =",I3," TO",I3," SECONDS",/7X,
     +"NUMBER OF BEAMS RECORDED =",I3)

      WRITE(1,1073)
 1073 FORMAT(7X,"PING",4X,"COMPASS",3X,"FIRST",3X,"HEADING",5X,
     +"TIME",/6X,"NUMBER",2X,"(DEGREES)",2X,"BEAM",3X,"(DEGREES)")

      DO 1076 I=1,IHDR(10)
      CALL HEADR(I,IHDR)
```

```
C > > > > COMPUTE THE TIME OF PING
      CALL PTIME(I, ICOM)

 1076 WRITE(1,1077) I,IHDR(3),IHDR(5),IHDR(6),KT1,KT2,KT3
 1077 FORMAT (8X,I2,6X,I4,6X,I2,6X,I4,5X,I2,":",I2,":",I2)

C > > > > ANALYZER DATA STATUS TABLE GENERATION
 1778 WRITE (2,1078)
 1078 FORMAT (/17X,"ANALYZED DATA STATUS TABLE"/
     +2X,"PING",3X,"CENTER BEAM",2X,"NUMBER BEAMS",3X,"NUMBER"
     +" OF",4X,"AVERAGE"/1X,"NUMBER",5X,"ANGLE",6X,"PROCESSED",
     +4X,"EXCEEDENCES",2X,"EXCEEDENCE")

      DO 1080 I=1,10
      IF (NRBMS(I) .EQ. 0 .AND. ITH(I) .EQ. 0) GO TO 1080
      WRITE (2,1079) I,MIDBM(I),NRBMS(I),ITH(I),
     +IAVGEX(I)
 1079 FORMAT (3X, I2, 8X, I3, 11X, I2, 10X, I4, 9X, I3)
 1080 CONTINUE

C > > > > DRAW GRID AROUND THESE TABLES
      CALL STGRD (IHDR(10),NRBMS,ITH)

C > > > > POSITION TO END OF PAGE
 1776 CALL TPLOT (0,0,0)
      CALL CHOUT (37B)
      WRITE (2,1777)
 1777 FORMAT (" ")
      IF (IMOD .EQ. 1) GO TO 1010
        GO TO 1210
```

```
C   **************************************************************
C   *  OFF-LINE DATA ANALYSIS & PLOT DISPLAY FUNCTION            *
C   **************************************************************

C > > > > CLEAR SCREEN AND DISPLAY ERROR MESSAGE
 1200 CALL CHOUT (33B)
      CALL CHOUT (14B)
      CALL D750
      WRITE  (2,1205)
 1205 FORMAT (/"**********  NO ANALYZED DATA AVAILABLE "
     +"  **********"/)
      KEY = 1

C > > > > ASK OFF-LINE MODE QUESTIONS
 1210 CALL ASKQ (KEY,1,4,20,IDATE)
      NFTSW = 3
      IF (KEY .EQ. 1)  GO TO 1030
      IF (ITH(IPNG) .EQ. 0)  GO TO 1200

C > > > > TRANSFER TO THE SELECTED OFF-LINE MODE
      GO TO (1030,1040,1050,1060,1040) KEY

C   ****************************************************
C   *  ANALYSIS OF PREVIOUSLY RECORDED DATA  (KEY = 1)  *
C   ****************************************************

C > > > > CHECK FOR HIGHEST PING REQUESTED FOR RECORDING
 1030 IFLAG = 1
      CALL HEADR (1,IHDR)

C >> DIAGONSTICS >>  SENSE SW. 3 (ON)
C     IF(ISSW(10) .LT. 0) WRITE(1, 9102) (ICOM(J), J = 51, 66)

      NHI = IHDR(10)
```

```
C > > > > ASK OFF-LINE ANALYSIS QUESTIONS
      WRITE (2,1031)
 1031 FORMAT (/5X,"**  ANSWER ANALYSIS QUESTIONS  **"/)
      ASSIGN 1110 TO IRET
        GO TO 500

C > > > > RE-ENTER FROM QUESTION ROUTINE
 1110 CONTINUE

C > > > > CONVERT START/STOP RANGE
      CALL RBINS(IFST, JSTRT, JEND, KOFFS, MIN, MAX, INCRX)

C > > > > CLEAR DATA STATUS VARIABLES
      DO 1303 I = IFST,ILST
       IAVGEX(I) = 0
       MIDBM(I)  = 999
       NRBMS(I)  = 0
 1303  ITH(I)    = 0

C > > > > OFF-LINE ANALYSIS LOOP
      DO 1038 IPNG=IFST,ILST

C > > > > TRANSFER TO SINGLE PING ANALYSIS
      ASSIGN 1115 TO IRET
        GO TO 600

C > > > > RE-ENTER FROM SINGLE PING ANALYSIS ROUTINE
1115  CONTINUE
      AVGEXC = SUMEXC / ITH(IPNG)
      IAVGEX(IPNG) = AVGEXC

C > > > > SAVE NUMBER OF EXCEEDANCES PER PING
      IDAT = ITH(IPNG)
1038  CONTINUE
C  ****************  END OFF-LINE ANALYSIS LOOP  ****************
```

```
      IPNG = IPNG - 1
      KTHR = 1
        GO TO 1090

C  **************************************************************
C  *  INDIVIDUAL BEAM DISPLAY  (KEY = 2)                        *
C  **************************************************************

C > > > > ASK INDIVIDUAL BEAM DISPLAY QUESTIONS
 1040 IFLAG = 4
      WRITE (2,1041)
 1041 FORMAT (/5X,"**  ANSWER INDIVIDUAL BEAM DISPLAY "
     +"QUESTIONS  **"/)
      ASSIGN 1090 TO IRET
        GO TO 500

C  **************************************************************
C  *  MULTI-PING DISPLAY  (KEY = 3)                             *
C  **************************************************************

C > > > > ASK MULTI-PING DISPLAY QUESTIONS
 1050 IFLAG = 2
      WRITE (2,1051)
 1051 FORMAT (/5X,"**  ANSWER MULTI-PING DISPLAY QUESTIONS  **"/)
      ASSIGN 1090 TO IRET
        GO TO 500

C  **************************************************************
C  *  FOUR PART DISPLAY  (KEY = 4)                              *
C  **************************************************************

C > > > > ASK FOUR PART DISPLAY QUESTIONS
 1060 IFLAG = 3
      WRITE (2,1061)
 1061 FORMAT (/5X,"**  ANSWER FOUR PART DISPLAY QUESTIONS  **"/)
```

```
      ASSIGN 1090 TO IRET
        GO TO 500

C   ***************************************************************
C   *  TRANSFER TO CORELOAD 7 FOR PLOT DISPLAY                    *
C   ***************************************************************

C > > > > RE-ENTER FROM QUESTION ROUTINE
 1090 CONTINUE

C > > > > CONVERT START/STOP RANGE, GET TIME OF IPNG
      CALL RBINS(IPNG, JSTRT, JEND, KOFFS, MIN, MAX, INCRX)
      CALL PTIME(IPNG, ICOM)

C > > > > TRANSFER TO CORELOAD 7 FOR PLOT DISPLAYS
      KLOAD = 7
        GO TO 800

C   *********************  LEAVE CORELOAD 3  *********************


C   ***************************************************************
C   *                                                             *
C   * QUESTION ROUTINE:  COLLECTS USER-ENTERED PARAMETES FOR USE  *
C   *                    IN THE OLAAS DATA ANALYSIS.              *
C   *                                                             *
C   *     NOTE: IFLAG VALUES:  0 = ON-LINE FUNCTION               *
C   *                          1 = OFF-LINE ANAYLSIS              *
C   *                          2 = MULTI-PING DISPLAY             *
C   *                          3 = FOUR PART DISPLAY              *
C   *                          4 = INDIVIDUAL BEAN DISPLAY        *
C   *                                                             *
C   ***************************************************************

C > > > > ENTER QUESTION ROUTINE & DETERMINE STARTING QUESTION
```

```
  500 NFTTAG = NFTSW
      NFTSW  = 5
      IQUEIN = IFLAG + 1
      GO TO (515, 501, 501, 503, 503) IQUEIN

C > > > > FIRST PING
 501  CALL ASKQ (IFST,1,NHI,1)

C > > > > LAST PING
      CALL ASKQ (ILST,IFST,NHI,2)
        IF (IFLAG .EQ. 1) GO TO 515
        IF (IFLAG .EQ. 2) GO TO 525

C > > > > DISPLAY PING NUMBER  (INDIVIDUAL BEAM OR FOUR PART)
  503 CALL ASKQ (IPNG,1,10,14)
        IF (IFLAG .EQ. 3) GO TO 510

C > > > > DISPLAY TYPE
  505 CALL ASKQ (KEY,2,5,15)
      IF (KEY .EQ. 2 .OR. KEY .EQ. 5) GO TO 525
      WRITE (2,507)
  507 FORMAT ("*********** PARAMETER ERROR ***********",
     +5X,"RE-ENTER VALUE")
      GO TO 505

C > > > > DISPLAY BEARING (FOUR PART DISPLAY)
  510 CALL ASKQ (JDIR,0,359,16)
        IF (IFLAG .EQ. 3) GO TO 525

C > > > > BUOY FREQUENCY
  515 LFRQ = 1
      IF (ISSW(15) .LT. 0)  CALL ASKQ (LFRQ,1,2,3)

C > > > > PULSE LENGTH
      LNGTH = 2
```

```
      IBS = 1024
      IF (ISSW(15) .LT. 0)  CALL ASKQ (LNGTH,2,5,4)
      IF(LNGTH .EQ. 5) IBS=2048
      NQUAL(1)=IBS
      NQUAL(2)=100B
      NQUAL(3)=16383
      NQUAL(4) = 45
      NQUAL(5) = 0
      CALL PUTQ(0,NQUAL)

C > > > > SIGNAL MODE (CW, FM-UP, OR FM-DOWN)
      CALL ASKQ (MODE,0,2,5)
        IF (MODE .GE. 1) GO TO 520

C > > > > CW DOPPLER BAND
      CALL ASKQ (IDPLR,1,4,6)

C > > > > CW NOTCH FILTER SETTING
      LCELL = ((10.0 * IBS)/666.67) + 0.5
      IF (ISSW(15) .LT. 0)  CALL ASKQ (LCELL,0,25,17)
        GO TO 525

C > > > > FM DOPPLER BAND
  520 CALL ASKQ (IDPLR,1,2,7)

C > > > > START RANGE
  525 CALL ASKQ (JSTRT,2,35,8)

C > > > > STOP RANGE
      CALL ASKQ (JEND,JSTRT,40,9)
        IF (IFLAG .EQ. 2 .OR. IFLAG .EQ. 4)  GO TO 530
        IF (IFLAG .EQ. 3)  GO TO 538

C > > > > RANGE RESOLUTION
      CALL ASKQ (INCRT,1,40,10)
```

```
C > > > > BEARING OF CENTER BEAM
  530 CALL ASKQ (JDIR,0,359,11)
        IF (IFLAG .EQ. 2) GO TO 538

C > > > > NUMBER OF BEAMS ANALYZED
      CALL ASKQ (KNUM,1,16,12)
        IF (IFLAG .EQ. 4) GO TO 538

C > > > > ANALYSIS THRESHOLD
  535 CALL ASKQ (IFRAC,0,10000,13)
        IF (IFLAG .LT. 3) GO TO 540

C > > > > PLOT THRESHOLD
  538 CALL ASKQ (KTHR,1,100,18)
        IF (IFLAG .GE. 2) GO TO 595

C >>>>>>> SET-UP FOR CW & FM DOPPLER >>>>>>>
  540 IF (MODE .EQ. 0) GO TO 575

C > > > > CALCULATE THE FM REPLICA
      LSWP = 0
      IF (MODE .EQ. 2) LSWP = 1
      BW = (IBS * 100.0)/(666.667 * LNGTH)
      IF (LFRQ .EQ. 1) GO TO 542
      IF (LSWP .EQ. 0) GO TO 545
      KSWP = 0
        GO TO 550
  545 KSWP = 1
        GO TO 550
  542 KSWP = LSWP

C > > > > SET SCALE FACTORS FOR FM REPLICA
  550 KQUAL(1) = IBS
      KQUAL(2) = 100B
```

```
      KQUAL(3) = 32767
      KQUAL(4) = 45
      CALL PUTQ(1,KQUAL)
      IF (KSWP) 560,560,555
  555 FS = 83.0 + (BW/2.0)
      SS = -100.0/LNGTH
        GO TO 565
  560 FS = 83.0 - (BW/2.0)
      SS = 100.0/LNGTH
  565 DT=1.0/666.66667
      DO 570 I = 0,(IBS-1)
      REPLI=COS(2.*3.14159*((FS*I*DT) + ((SS/2.0)*((I*DT)**2))))
      J = REPLI * 32767.0
      CALL PUTW (1,I,J,0)
  570 CONTINUE
      CALL PUTQ (0,NQUAL)

C > > > > DOPPLER EDITING FOR FM
      JDIV = 2
      NROT = 64
      IF (IDPLR .EQ. 2) JDIV = 4
        GO TO 580

C > > > > DOPPLER EDITING FOR CW
  575 JDIV = 4
      IF (IDPLR .EQ. 1) JDIV = 2
      NROT = 128
      IF (IDPLR .EQ. 2) NROT=192
      IF (IDPLR .EQ. 3 .AND. LFRQ .EQ. 1) NROT=256
      IF (IDPLR .EQ. 4 .AND. LFRQ .EQ. 2) NROT=256
  580 IBLK = 0
      IF (LNGTH .EQ. 5) NROT = NROT*2
      NEWBS = IBS/JDIV

C > > > > SET-UP TO RETURN
```

```
  595 SUMEXC = 0.0
      NFTSW  = NFTTAG

C > > > > RETURN TO CALLER
        GO TO IRET

C  ###################   END QUESTION ROUTINE   ####################


C  ****************************************************************
C  *                                                              *
C  *  ANALYSIS ROUTINE:   ANALYZES BEAMFORMER DATA AS SPECIFIED   *
C  *                      BY THE OLAAS DATA ANALYSIS PARAMETERS,  *
C  *                      FOR A SINGLE PING.                      *
C  *                                                              *
C  ****************************************************************

C > > > > ENTER ANALYSIS ROUTINE FOR SINGLE PING ANALYSIS
  600 CONTINUE
      NPAGE = 1
      KTDAT = 0
      KOUNTB= 0
      NRBLK = 0
      LSTDAT= 0
      LA1   = -4
      SUMEXC= 0.0

C > > > > TEST FOR FM MODE
      IF (MODE .GE. 1) GO TO 798

C > > > > COMPUTE PARAMETERS FOR THE CW REVERB NOTCH FILTER
      ILLIM = IBS / 4 - LCELL
      IULIM = IBS / 4 + LCELL
      II = 0
798   CONTINUE
```

```
      CALL HEADR(IPNG,IHDR)

C > > > > COMPUTE THE TIME
      CALL PTIME(IPNG, ICOM)
      KMPAS=IHDR(3)

C > > > > CORRECTED DIRECTION MTRUE
      AIBM=((JDIR-KMPAS)/22.5)+.5
      IF(AIBM .GT. 16.)AIBM=AIBM-16.
      IF(AIBM.LT.1.)AIBM=AIBM+16.
      IBM=AIBM
  241 IBEAM = IBM - (KNUM/2)
      IF(IBEAM .LT. 0)IBEAM = IBEAM + 16
      KOUNT=1

C > > > > COMPUTE DIRECTION OF BEAM
      DIR=(IBEAM * 22.5) + KMPAS -22.5
      IF(DIR .GE. 360.0)DIR=DIR-360.0

C  **************************************************************
C  *  START OF BEAM LOOP                                        *
C  **************************************************************

  228 CONTINUE
      KBEAM=IBEAM+1
      IF(KBEAM .GT. 16)KBEAM = KBEAM - 16

C > > > > CHECK IF BEAM WAS RECORDED
      IF(IHDR(8) .EQ. 16) GO TO  211
      DO 210 I = IHDR(5),(IHDR(8) + IHDR(5) - 1)
      KQ = I
      IF(KQ .GT. 16) KQ = KQ - 16
      IF(KBEAM .EQ. KQ ) GO TO 211
  210 CONTINUE
      WRITE(1, 207) KBEAM
```

```
  207 FORMAT(" BEAM ",I2," WAS NOT RECORDED")

C > > > > RETURN TO CALLER
        GO TO IRET

C  ##################  EXIT ANALISIS ROUTINE  ####################

  211 CONTINUE
      DIR=DIR+22.5
      KDIR=DIR+.5
      IF(KDIR .GE. 360)KDIR = KDIR - 360
      MIN = MAX0(MIN, 0)
      MAX = MIN0(MAX, IFIX(IHDR(12) / 0.192) )

C  ***************************************************************
C  *  BEGIN DATA COLLECTION LOOP                                  *
C  ***************************************************************

  250 DO 900 INDEX=MIN,MAX,INCRT

C >> DIAGONSTICS >>  SENSE SW. 7 (ON)
      IF(ISSW(7) .LT. 0)  GO TO 97

C > > > > SET SCALE FACTORS  (10**1,LINEAR,RECT,TIME)
      NQUAL(1)=IBS
      CALL PUTQ(0,NQUAL)

C > > > > OBTAIN TIME-DOMAIN BEAMFORMER DATA

C >> DIAGONSTICS >>  SENSE SW. 9 (ON)
C     IF(ISSW(9) .LT. 0)   WRITE(1, 991)IPNG,KBEAM,INDEX,IBS,IERR

      CALL RDATA(IPNG,KBEAM,INDEX,IBS,IERR)

C > > > > COMPUTE VARIABLE THRESHOLD HERE
```

```
  330 IF(IERR .NE. 0)  GO TO 99

C > > > > IF F.M., MULTIPLY BY BLOCK 1
      CALL ENDIO

C >> DIAGONSTICS >>  SENSE SW. 8 (ON)
C     IF(ISSW(8) .LT. 0) WRITE(1, 991)KDIR,IPNG,KBEAM,INDEX,
C    +IBS,IERR

C > > > > TEST FOR CW MODE
      IF (MODE .EQ. 0) GO TO 331

C > > > > SET DATA BLOCK 1 QUALIFIERS FOR FM
      KQUAL(1) = IBS
      KQUAL(2) = 100B
      KQUAL(3) = 32767
      KQUAL(4) = 45
      CALL PUTQ(1,KQUAL)

C > > > > MULTIPLY DATA BLOCK 1 BY 0
      CALL KYBD(025040B,1)

C > > > > HANN DATA BLOCK 0
  331 CALL KYBD(2HH1,0)

C > > > > TAKE FOURIER TRANSFORM OF DATA BLOCK 0
  332 CALL KYBD(2HF ,0)

C > > > > CONJUGATE MULTIPY OF DATA BLOCK 0
      CALL KYBD(2H*-,0)
      CALL ENDIO

C > > > > SKIP REVERBERATION FILTERING IF FM
      CALL ENDIO
```

```
C >> DIAGONSTICS >>  SENSE SW. 2 (ON)
C     IF(ISSW(2) .LT. 0) WRITE(1, 9793)IBS,LCELL,ILLIM,IULIM,NROT
C9793  FORMAT("IBS,LCELL,ILLIM,IULIM,NROT",5X, 5I6)

C > > > > TEST FOR FM MODES
      IF (MODE .GE. 1) GO TO 793
      DO 8222 L = ILLIM,IULIM
8222  CALL PUTW(0,L,0,II)
793   CONTINUE

C > > > > ROTATE SPECTRUM NROT CELLS TO THE LEFT
      IF (MODE .GE. 1 .AND. IDPLR .EQ. 1) GO TO 8443

C > > > > SHIFT DATA BLOCK 0 FOR FM HALF DOPPLER
      CALL KYBD(057440B,0,NROT)
      CALL ENDIO
8443  CALL GETQ(0,LQUAL)
      LQUAL(1)=NEWBS
      CALL PUTQ(0,LQUAL)

C > > > > OBTAIN ABSOLUTE MAGNITUDE
      CALL ENDIO

C >> DIAGONSTICS >>  SENSE SW. 5 (ON)
C     IF(ISSW(5) .LT. 0) WRITE(1,991)KBEAM,INDEX,LQUAL

C > > > > SEARCH CELLS IN BLOCK FOR THRESHOLD EXCEEDENCES
      CALL GETQ(0,LQUAL)

C >> DIAGONSTICS >>  SENSE SW. 10 (NOT ACTIVE)
C     IF(ISSW(10) .LT. 0) WRITE(1, 991)LQUAL

      CALIB = 1.0 * LQUAL(3)
      IEXP = ISHFT(LQUAL(2),6)

```

```
C > > > > CALCULATE THE ANALYSIS THRESHOLD
      RTHR = IFRAC
      IF (ISSW(14) .GE. 0) GO TO 9100

C > > > > STORE FOURIER DATA BLOCK 0 IN NTEMP
      NTEMP = (4096 / NEWBS) + 1
      CALL KYBD (2HX>,NTEMP)

C > > > > INTEGRATE FOURIER DATA BLOCK 2
      CALL KYBD (2H$ ,NTEMP,1,((NEWBS/2)-1))

C > > > > GET THE AVERAGE THRESHOLD
      CALL GET (NTEMP,((NEWBS/2)-1),PWR,DATAI)
      RTHR = (IFRAC * PWR / ((NEWBS/2)-2)) / 100.0

C >> DIAGONSTICS >>  SENSE SW. 6 (ON)
C        NEWBLO = NEWBS/2
C        IF (ISSW(6) .LT. 0) WRITE (2,9101) IFRAC,NEWBLO,PWR,RTHR
C9101    FORMAT ("IFRAC",5X,"NEWBS/2",5X,"PWR",10X,"RTHR"
C     +/,I5,5X,I5,5X,F8.6,5X,F8.6)

 9100 IF (ISSW(14) .GE. 0) RTHR = RTHR * 1E-4
      THRES =(RTHR * 32767.0**2)/(CALIB * 10.0**IEXP)

C >> DIAGONSTICS >>  SENSE SW. 6 (NOT ACTIVE)
C        IF(ISSW(6) .LT. 0) WRITE(1, 9087)THRES,CALIB,RTHR,IEXP
C9087       FORMAT(1X,3F15.5, 1I9)

      IF(THRES .GT. 32767.) THRES = 32767.
      JTHR = THRES + 0.5
      NQUAL(1) = 2048

C   ************************************************************
C   *  BEGIN DOPPLER LOOP                                      *
C   ************************************************************
```

```
      DO 95 KDOP = 0,((NEWBS/2)-1)
      CALL GETW(0,KDOP,IDAT1,IDAT2)
991    FORMAT(2X, 10I6)

C > > > > TEST FOR THRESHOLD EXCEEDENCE
      IF(IDAT1 .LT. JTHR ) GO TO 95
90    KTDAT = KTDAT + 1
      IDATC = IFIX(100.0 * (IDAT1 - JTHR) / IDAT1 + 0.5)
      SUMEXC = SUMEXC + 1.0 * IDATC

C > > > > GET LOCATIONS IN BLOCK 3, STORE 3 DATA POINT VALUES
      LA1 = 4 * (KTDAT-1)
      CALL PUTQ(3, NQUAL)
      CALL PUTW(3,LA1,    KDIR, II)
      CALL PUTW(3,LA1+1,INDEX, II)
      CALL PUTW(3,LA1+2,IDATC, II)
      CALL PUTW(3,LA1+3, KDOP, II)
      ASSIGN 95 TO KRTN
      IF(KTDAT .GE. 512)          GO TO(98,98,97)NPAGE
95     CONTINUE

C *********************  END OF DOPPLER LOOP *********************

900   CONTINUE

C  ****************  END OF DATA COLLECTION LOOP  ****************
C >> DIAGONSTICS >>  SENSE SW. 10 (ON)
C     IF(ISSW(10) .LT. 0) WRITE(1, 9900)
C9900   FORMAT(" END OF RANGE INDEX LOOP")

       IBEAM = IBEAM + 1
       IF(IBEAM .GE. 16) IBEAM = IBEAM - 16
       KOUNTB = KOUNTB + 1
       IF(IBEAM .LE. IBM+1) MIDBM(IPNG) = KDIR
```

```
      NUSED  = ITH(IPNG) + KTDAT

C >> DIAGONSTICS >>  SENSE SW. 3 (ON)
      IF(ISSW(3) .LT. 0) WRITE(1, 9096)KDIR,NUSED,IPNG
9096   FORMAT(I4," DEGREE BEAM COMPLETE.  ",I4," DATA POINTS USED
     + " (OF 1536) FOR PING",I3)

      IF(KOUNTB .LT. KNUM)  GO TO 228

C  ********************  END OF BEAM LOOP  ********************

   97 LSTDAT = 999
      LA1    = MIN0(2044, LA1+4)
      NRBMS(IPNG) = KOUNTB
      CALL PUTQ(3, NQUAL)
      CALL PUTW(3, LA1, 999,II)

C > > > > STORE CORE DATA BLOCK 3 TO DISC.  (PING 1 BEGINS
C          IN BLOCK 1).
   98 NRBLK = 3 * (IPNG - 1) + NPAGE

C >> DIAGONSTICS >>  SENSE SW. 3 (ON)
      IF(ISSW(3) .LT. 0) WRITE(1, 9098) NRBLK
 9098 FORMAT(" WRITING TO DISC DATA BLOCK ", I3)

      CALL PUTQ(3, NQUAL)
      CALL KYBD(2HMS, 31, NRBLK)
      CALL KYBD(2HMS, 21, 3)
      CALL KYBD(2HCL, 3)
      NPAGE = NPAGE + 1
      ITH(IPNG)  = ITH(IPNG) + KTDAT
      NRBMS(IPNG)= KOUNTB
      KTDAT = 0
      LA1   =-4
```

```
C >> DIAGONSTICS >>  SENSE SW. 10 (ON)
C       IF(ISSW(10) .GE. 0)  GO TO 622
C > > > > VERIFY DATA JUST WRITTEN ON DISC
C        WRITE(1, 9998)
C        WRITE(1, 987) NQUAL
C987     FORMAT(" NQUAL =", 5I8)
C       CALL PUTQ(3,NQUAL)
C        CALL KYBD(2HMS, 31,NRBLK)
C        CALL KYBD(2HMS, 11,3)
C        DO 620  J = 0, 100, 4
C         CALL GETW(3, J,   N1, II)
C         CALL GETW(3, J+1, N2, II)
C         CALL GETW(3, J+2, N3, II)
C         CALL GETW(3, J+3, N4, II)
C         WRITE(1, 9620) N1, N2, N3, N4, J
C9620     FORMAT(1X, 5I10)
C       IF(N1 .EQ. 999)    GO TO 621
C620    CONTINUE
C621    WRITE(1, 9998)

C > > > > TEST FOR LAST DATA & RETURN TO CALLER
 622    IF(LSTDAT .EQ. 999)  GO TO IRET

C  ##################  EXIT ANALYSIS ROUTINE  ####################

C > > > > TEST FOR 1536 DETECTIONS
        CALL PUTW(3, 0, 999, II)
        IF(NPAGE .LE. 3)          GO TO KRTN
        WRITE(1, 990) IPNG
 990    FORMAT(/"STOPPED AFTER 1536 DETECTIONS ON PING ",I4)

C > > > > RETURN TO CALLER
          GO TO IRET

C  ##################  EXIT ANALYSIS ROUTINE  ####################
```

```
   99  WRITE(1, 999)
  999  FORMAT(" NO DATA RETURNED FROM SUBROUTINE RDATA")
         GO TO 97

C  ###################  END ANALYSIS ROUTINE  ###################


C  ****************************************************************
C  *  SAVE ICOM IN DISC BLOCK 0 AND EXIT                          *
C  ****************************************************************

C >> DIAGONSTICS >>  SENSE SW. 1 (ON)
  800  IF(ISSW(1) .LT. 0)   PAUSE 2
       NFTTAG = NFTSW
       CALL PUTQ(3, NQUAL)
       DO 801  J = 1, 127
        CALL PUTW(3, J, ICOM(J), II)
   801 CONTINUE
       CALL KYBD(2HMS,31,0)
       CALL KYBD(2HMS,21,3)

C >> DIAGONSTICS >>  SENSE SW. 10 (ON)
       IF(ISSW(10) .LT. 0) WRITE(1, 9801)KLOAD, NFTSW, NQUAL
 9801    FORMAT(/," CORELOAD",I2, "  NFTSW=",I2, "  NQUAL ",5I7)
       IF(ISSW(10) .LT. 0) WRITE(1, 9999)ICOM
 9999    FORMAT("03", 10I7)

C >> DIAGONSTICS >>  SENSE SW. 10 (ON)
       IF(ISSW(10) .LT. 0) WRITE(1, 9998)
       IF(ISSW(10) .LT. 0) PAUSE 2000

   802 CALL ENDIO
       CALL CDUMP(23B)
   805 N = KLOAD + 1
```

```
C > > > > TRANSFER TO SELECTED CORELOAD
      CALL CLOAD(KLTRK(N), KLADR(N) )

C  ***************  TRANSFER TO SELECTED CORELOAD  ***************
      END


      SUBROUTINE STGRD (NRCDS,NRBMS,ITH)
C
C ****************************************************************
C *  STGRD SUBROUTINE DRAWS GRIDS AROUND THE DATA STATUS TABLES *
C ****************************************************************

      DIMENSION NRBMS(10), ITH(10)

C > > > > GRID FOR RECORDED DATA STATUS TABLE
 7000 CALL TPLOT (0,70,698)
      CALL TPLOT (1,728,698)
      CALL TPLOT (0,70,632)
      CALL TPLOT (1,728,632)
      CALL TPLOT (0,70,588)
      CALL TPLOT (1,728,588)
      IYHL = 588
      DO 7005 I=1,NRCDS
        IYHL = IYHL - 22
        CALL TPLOT (0,70,IYHL)
        CALL TPLOT (1,728,IYHL)
 7005 CONTINUE
      CALL TPLOT (1,728,720)
      CALL TPLOT (1,70,720)
      CALL TPLOT (1,70,IYHL)
      CALL TPLOT (0,182,632)
      CALL TPLOT (1,182,IYHL)
      CALL TPLOT (0,326,632)
```

```
      CALL TPLOT (1,326,IYHL)
      CALL TPLOT (0,434,632)
      CALL TPLOT (1,434,IYHL)
      CALL TPLOT (0,588,632)
      CALL TPLOT (1,588,IYHL)
      CALL CHOUT (37B)

C > > > > DRAW GRID FOR ANALYZED DATA STATUS TABLE
 7010 IYHLA = IYHL - 45
      IYHL = IYHL - 89
      CALL TPLOT (0,0,IYHLA)
      CALL TPLOT (1,840,IYHLA)
      CALL TPLOT (0,0,IYHL)
      CALL TPLOT (1,840,IYHL)
      DO 7015 I=1,10
        IF (NRBMS(I) .EQ. 0 .AND. ITH(I) .EQ. 0) GO TO 7015
        IYHL = IYHL - 22
        CALL TPLOT (0,0,IYHL)
        CALL TPLOT (1,840,IYHL)
 7015 CONTINUE
      CALL TPLOT (1,840,IYHLA+22)
      CALL TPLOT (1,0,IYHLA+22)
      CALL TPLOT (1,0,IYHL)
      CALL TPLOT (0,112,IYHLA)
      CALL TPLOT (1,112,IYHL)
      CALL TPLOT (0,294,IYHLA)
      CALL TPLOT (1,294,IYHL)
      CALL TPLOT (0,490,IYHLA)
      CALL TPLOT (1,490,IYHL)
      CALL TPLOT (0,672,IYHLA)
      CALL TPLOT (1,672,IYHL)
      CALL CHOUT (37B)
C > > > > RETURN TO CALLER
      RETURN
C  ********  END OF STATUS TABLE GRID DRAWING SUBROUTINE  ********
```

```
      END


      SUBROUTINE RBINS(IPNG,JSTRT,JEND,KOFFS,MIN,MAX,INCRX)

C ***********************************************************************
C *  THIS ROUTINE CONVERTS START AND STOP RANGE IN KILOYARDS             *
C *  TO MIN AND MAX RANGE INDICES FOR PLOTS.                             *
C *  START AND END TIMES (FROM END OF XMIT. PULSE) ARE OBTAINED           *
C *  FROM CALLING HEADER DATA STORED DURING DATA COLLECTION.             *
C *  INCRX IS THE LENGTH OF A RANGE BIN IN PLOT POINTS.                  *
C ***********************************************************************

      DIMENSION IHDR(16)

      CALL HEADR(IPNG,IHDR)
      KOFFS = IHDR(11)
      MIN   = IFIX( (JSTRT * 3000./2500. - KOFFS)/0.192 + 0.5)
      MAX   = IFIX( (JEND  * 3000./2500. - KOFFS)/0.192 + 0.5)
      INCRX = IFIX(896.0 / (MAX - MIN) + 0.5)

C >> DIAGONSTICS >>  SENSE SW. 10 (ON)
C       IF(ISSW(10) .LT. 0)  WRITE(1, 7110)MIN,MAX,INCRX,IPNG,
C    *                                     JSTRT,JEND,KOFFS
C7110  FORMAT("MIN, MAX, INCRX =", 7I6)
C       IF(ISSW(10) .LT. 0) PAUSE 10

C > > > > RETURN TO CALLER
        RETURN

C ************  END OF RANGE BINS CONVERT SUBROUTINE  ***********
      END


      SUBROUTINE PTIME(IPNG, ICOM)
```

```
C     ***************************************************************
C     *  THIS ROUTINE CONVERTS PING TIME FROM BCD UNITS (OBTAINED)  *
C     *  FROM THE HEADER WRITTEN DURING DATA COLLECTION) INTO       *
C     *  BINARY FORM.  RESULTS ARE STORED IN PROPER ICOM LOCATIONS. *
C     ***************************************************************

      DIMENSION ICOM(127), IHDR(16), IBCD(6), MASK(4)

      DATA MASK/17B,360B,7400B,170000B/

      CALL HEADR(IPNG, IHDR)
C > > > > HOURS
      IBCD(5) = IAND(MASK(1), IHDR(1) )
      IBCD(6) = IAND(MASK(2), IHDR(1) )
      ICOM(14)= IBCD(6) / 20B * 10 + IBCD(5)
C > > > > MINUTES AND SECONDS
      DO 10  I = 1, 4
   10 IBCD(I)  = IAND(MASK(I), IHDR(2) )

      ICOM(15) = IBCD(4) / 10000B * 10 + IBCD(3)/400B
      ICOM(16) = IBCD(2) /    20B * 10 + IBCD(1)

C > > > > RETURN TO CALLER
        RETURN

C     $$$$$$$$$$$$$$$$  END OF PING TIME SUBROUTINE  $$$$$$$$$$$$$$$$


C     $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C     $$$$$$$  END OF ANALYSIS & SYSTEM EXECUTIVE SUBROUTINE  $$$$$$$
C     $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
      END
      END$
```

OLAAS DATA COLLECTION PROGRAM - Y0001

1. FUNCTION. Y0001 collects, demultiplexes, and stores selected beam data on the disc.

2. CONSTRAINTS. None.

3. CALLING SEQUENCE. N/A

4. DESCRIPTION OF INPUT. See Figure 3 for description of Beam Former input and method. See Figure 4 for description of Buoy Data input and method. See Figure 5 for description of Time Code Generator input and method.

5. DESCRIPTION OF OUTPUT. All output is to the disc. See Figure 6 for format.

6. FILES USED. N/A

7. ERRORS AND RESTRICTIONS. The amount of data which can be written is limited by the Track Allocation Table. If an attempt is made to write beyond the allocated tracks, an irrecoverable error halt will occur.

8. COMPUTER OPERATOR INSTRUCTIONS. See general system operating instructions.

9. DESCRIPTION OF PROCESSING. Y0001 must keep track of whether the current execution is the first execution or not. When it is not the first execution, it has been read in from disc tracks to which it was written out following its previous execution, therefore, the states of all switches, counters, etc., have been saved. At (1), assuming this is the initial execution (ping 1), the not first time switch NFTSW will be zero. NFTSW will be set non-zero (not first time) and the operator parameters N, L, K, M, D, S are set to illegal values (-1). INPUT is called to obtain these parameters from the operator. The variables #BMS, TOTPG, and PING are initialized. At (2) the number of output blocks per ping is computed. This is the number of output buffer writes, consisting of 4 sectors per beam written. As the input data rate is 10667 Hz for 16 beams, the number of blocks expected during the ping can be computed from N, the amount of record time. The rounded, complemented result is stored in BLOCK and is used by the data retrieval routine to locate data on the disc. This number is rounded up to ensure that at least N seconds of data are recorded. The complemented result is stored in BLOCK and is used to time the recording process as well as used by the data retrieval routine to locate data on the disc.

At (3) the initialization of disc access pointers begins. One of the variables involved is head base, HEADB, which is the number of the lower head for the disc being used. This is either incremented or not according to which surface is being used. HEADB, TRAK, the current track, and MXTRK, the last track, are initialized for the lower disc. This means setting them to 2, FTLD, and LTLD, respectively. FTLD and LTLD are the first and last tracks for the lower disc as specified in the Track Allocation Table. MXTRK (from LTLD) is checked. If zero, no lower disc space has been allocated and the three variables are set to 0, FTUD, and LTUD, respectively, for the upper disc. If MXTRK is zero this time, no disc space at all has been allocated and an error exit is taken. LTUD is set to zero to indicate that the upper disc is currently in use. For either disc, current sector, SECT, is initialized to zero. All of these variables are calculated only once, on the first execution.

At (4) the logic flow from the not first time case rejoins us. A zero (NOP) is stored in cell 6. This is necessary because an interrupt request is pending from the core loading process, and will cause an interrupt to occur as soon as interrupts are enabled. There appears to be no way to handle this other than allowing it to occur and returning immediately to the interrupted program. The NOP does this. The block counter, BCTR, is set from BLOCK. SWAIT is called to wait for the transmit gate. After is occurs, the record delay, K, is multiplied by 10 to convert it to 0.1 second periods. This is necessary to increase the resolution of the delay. STDLY is called to initiate the counting of a period of K seconds. It returns immediately, leaving interrupts enabled and counting time base generator (TBG) interrupts as they occur. TDIN is called to input the time from the Time Code Generator and this data (2 words) is stored in the output buffer.

At (5) BDIN is called to input the 72 word Buoy Data block from the Avionics package. Of this, only the compass word, word 55, is used. The compass word is the heading of beam 1 in Gray code as a fraction of a revolution in bits 15-8. This is converted to weighted binary, then to degrees, then stored in ANGL1 and in the output buffer. The number of the first beam to be stored is then computed from the operator's desired direction, D, number of beams, L, and ANGL1. The equation is:

$$\text{FSTB\#} = \frac{\text{D-ANGL1}}{22.5} + 1 - \frac{\text{L-1}}{2}$$

The D-ANGL1 portion must be converted to positive degrees before proceeding. The result of the entire equation, a beam number, must be rounded before fixing. If the result is negative or zero, it is converted to a positive number by adding 16 (one revolution). Likewise, if the result is greater than 16, 16 is subtracted. The final result is stored in first beam number, FSTB#, and in the output buffer.

At (6) the angle of the first beam recorded is computed:

(FSTB#-1) * 22.5 + ANGL1

The result is rounded, corrected by subtracting $360^{o}$ if greater then $360^{o}$, and stored in the output buffer. The remaining header data is stored in the output buffer. The input buffer pointers are switched in preparation for Beam Former input.

At (7) DELAY is called to wait until the delay initiated by STDLY is over. BFIN is called to initiate input of 8192 words of Beam Former data. BFIN returns control immediately, so that the following processing takes place while input continues. Interrupts are turned off. WDISC is called to output the header record. Interrupts are turned back on and a delay of 1.2 seconds is begun. If this delay times out, this is an error condition, as the Beam Former completion interrupt should have occurred within this time.

When the Beam Former completion (DMA) interrupt occurs, control is transferred to location BFDUN, at (9). There the input buffer pointers are switched. BCTR is incremented and if not zero, BFIN is called again to start the next 8192 words of Beam Former input. If BCTR is zero, BFHLT is called to stop Beam Former input. In either case, remembering that BFIN returns control immediately, interrupts are turned off and demultiplexing (formatting) begins.

Formatting, at (10), is described much better by the flow chart than it can be in text, therefore the user is directed to the flow chart for details.

When formatting has been completed, there will be between 512 and 8192 words in the output buffer, depending upon the number of beams requested. At (11) the number of words in the buffer is computed (L*512) and that number of words is output by calling WDISC. A check is made on whether a Beam Former completion interrupt is waiting. If it is, the formatting and writing process took too long and consequently sync has been lost, so an error exit is taken. Otherwise, interrupts are turned on. BCTR is checked. If nonzero, control is transferred back to (8) to wait for Beam Former completion. If BCTR is zero, all requested data for this ping has been collected and the Beam Former has already been turned off. Ping number is incremented, the current core load is swapped out and the data analysis core load is read in. No check is made on whether PING has reached the limit TOTPG. It is the responsibility of the data analysis core load not to return control if all required data collection is finished.

Subroutines STDLY, TBGI, and DELAY work together. STDLY sets counter TCTR to the complement of the number of 0.1 second intervals passed to it in the calling sequence. It sets up the trap cell with a JSB through a base page

pointer to TBGI, starts the TBG at 0.1 second intervals, turns on interrupts, and returns. TBGI is entered when a TBG interrupt occurs. It turns off interrupts, saves the registers, and inputs the TBG status. If non-zero, a TBG tick has been missed, and an error exit is taken. Otherwise, TCTR is incremented. If TCTR is then zero, the desired count has been reached and the TBG is turned off with a CLC. Otherise, it is re-enabled with a CLF. In either case, the registers are then restored, interrupts are turned on, and control is returned to the point of interruption. Subroutine DELAY is called to determine the state of TCTR. If zero upon entry, an error exit is taken, as the count was exhausted at an unknown time in the past. Otherwise, the program loops until TCTR is zero and then returns.

Subroutine WDISC writes data as a sequential access file on the disc. It assumes that TRAK, SECT, and HEADB are set to specify the starting location on the disc. HEADB will always show the lower head for the disc in use; SECT varies from 0-47 to cover both surfaces. WDISC sets the core address and count into WPTR and WCTR from the calling parameters. At WDIS1 it computes AA, the number of words to be written on the current track. This will be the minimum of WCTR or the number of words remaining on the track. WRITD is called and will write AA words starting at TRAK and SECT. Next, SECT is updated. Since all write requests must be a multiple of 128 words (one sector), all that is required is to add AA/128 to SECT. If AA/128 is not an integer, an error path is taken. If the result is 48, SECT is reset to zero and TRAK is incremented by one. SECT+AA/128 will not be greater than 48 because AA was chosen to write only to the end of the current track. If TRAK is updated, it is compared to MXTRK. If greater, switching to the upper disc is called for, and TRAK and MXTRK are initialized from FTUD and LTUD. If MXTRK is now zero, either the upper disc was already being used, or no space was allocated on the upper disc, and an error exit is taken. Otherwise, LTUD is set to zero to indicate upper disc in use, and HEADB is set to zero. At WDIS2, WPTR and WCTR are updated by the number of words written, AA. If WCTR is non-zero, the all data has not been written and control is transferred back to WDIS1 to update AA and write additional data. This continues until WCTR is zero and then control is returned to the caller. Upon return, TRAK, SECT, and HEADB are properly set to continue writing from the point at which writing was discontinued.

WRITD, called by WDISC, first converts SECT (range 0-47) and HEADB to HDSEC, the control word for head and sector which uses sector in the range 0-23. The conversion consists of using head and sector as is if sector is less than 24, and otherwise, head plus one and sector less 24. This is then shifted into the bit positions used by the disc controller. DMA is set up to write AA words starting at the location in WPTR. The remainder of the disc processing is as specified in the disc interface manual with the exception that DMA 7 is used. DMA 7 is used because DMA 6 is in use for input during disc output. If status following the write is bad, an error path is taken.

SWAIT waits for the transmit gate, or S interrupt. To do so, it disables interrupts, performs a set control - clear flag on the time code generator interface, to which the S interrupt is connected, and loops until flag is set on the time code generator interface. SWAIT then clears control on the interface, turns interrupts back on and returns.

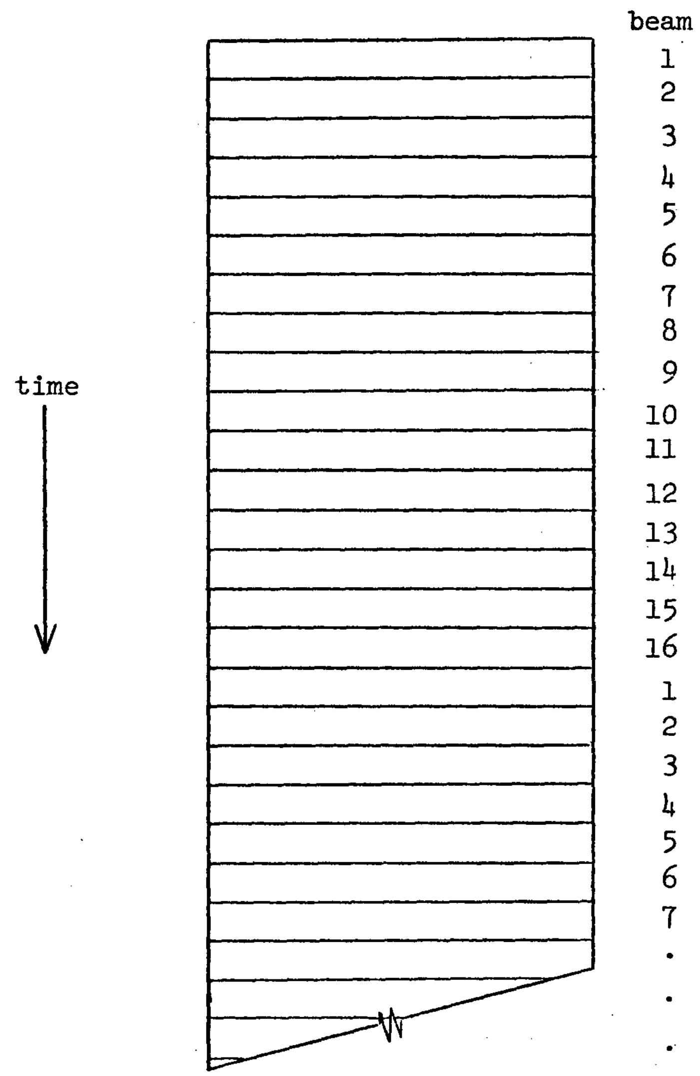
BFIN starts Beam Former input. It first verifies that the number of words called for in the calling sequence is a multiple of 16. If not, an error exit is taken. It sets up a JSB BFIT through a base page linkage in the DMA 6 interrupt trap cell, and an error halt in the Beam Former trap cell. The error halt is necessary to detect any delay in processing the DMA completion interrupt. DMA 6 is initialized for input of the number of words given in the calling sequence, the interface is enabled, DMA is started, and the Beam Former is started by outputting a 1 to the interface. This order is important, as otherwise the input does not dependably start with beam 1. Once input is started, control is returned to the caller. One of the calling parameters is the location to which control is to be transferred when input is complete. The immediate return to the caller is what allows the double buffering of input. As there are only about 93 microseconds between Beam Former words, and it is essential not to miss any, it would otherwise not be possible to do the formatting and output between BFIN calls. As it is, there is just enough time after a completion interrupt to start the next input request before the next word arrives.

BFIT processes the Beam Former DMA completion interrupts. It turns interrupts off, resets DMA 6, and branches to the completion address specified in the BFIN call. BFHLT is called from the main program to stop Beam Former input at the end of a ping. To do so, it outputs a zero to the interface, thus readying it for the synchronizing 1 on the next ping, clears control to prevent further interrupts, and returns.

BDIN inputs the 72 word Buoy Data record to the address specified in the calling sequence. The input method is the same as BFIN, but the word count is constant and control is not returned until input is complete. Therefore the use of interrupts is not required and all processing is performed with interrrupts off and using the skip-on-flag-set method to detect completion. When done, a zero is output to the interface, control is cleared on the interface and DMA, interrupts are turned back on, and control is returned to the caller.

TDIN inputs the time from the Time Code Generator. As there is no signal which indicates when the output is not changing, the input method is to look for two consecutive identical samples. A brief analysis will show that if the equipment is functioning properly, two consecutive identical samples must occur within four samples. TDIN simply samples until identical samples are found, or the count of four is reached, at which point an error exit is

taken. Subroutine TIN performs the actual sampling. It outputs a 1 in bit zero, which gates hours into the input register, and reads this into the A register. It then outputs a 1 in bit 1, which gates minutes and seconds into the input register, and reads this into the B register. Both registers are complemented (as the inputs are inverted) and control is returned to TDIN with hours in A and minutes and seconds in B.




When the level of the Beam Former Interface Card output bit 0 changes from 0 to 1, the next input word will be beam 1. Input data rate is 10667 words per second.

Figure 3. Beam Former Input Format

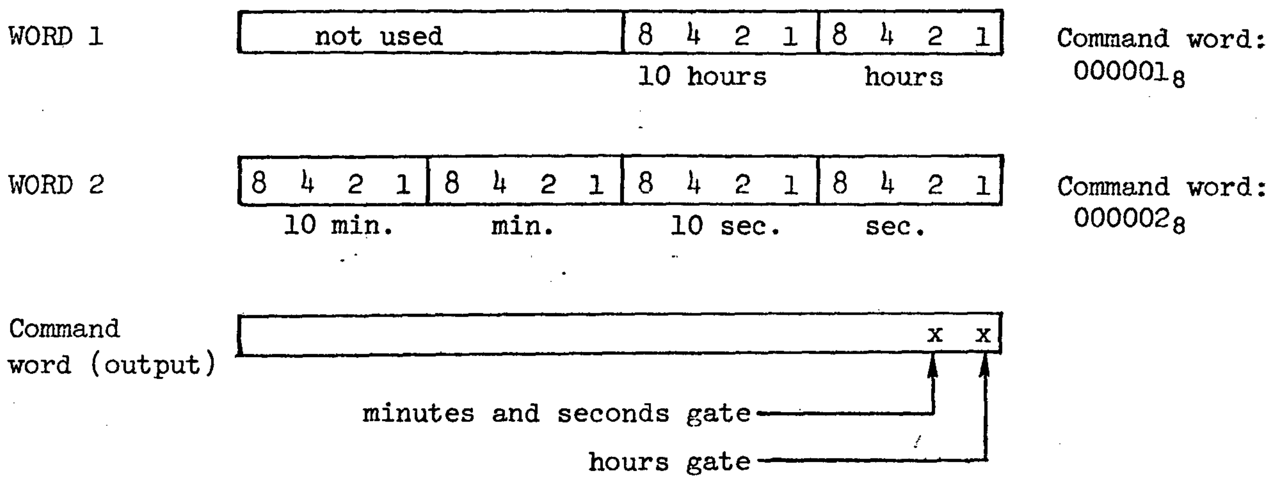
A data frame consists of 18 8-bit binary coded words, with the following format:

| Frame/Word | | | | | | |
|---|---|---|---|---|---|---|
| 1 | Sync 1 | Sync 2 | Data 1 | Data 2 | - - - - - - | Data 16 |
| 2 | Status | Blank | Data 1 | Data 2 | - - - - - - | Data 16 |
| 3 | AGC #1 | AGC #2 | Data 1 | Data 2 | - - - - - - | Data 16 |
| 4 | Compass | Parity | Data 1 | Data 2 | - - - - - - | Data 16 |
| 5 | Back to 1 (4 minor frames = 1 major frame) | | | | | |

The third word in the frame is the sample from the reference stave. The fourth word is the stave 2 sample, and so on in serial order up to the 18th word with stave 16. Staves 2 through 16 follow in clockwise order from the reference stave on the array, when viewed from the top of the array.

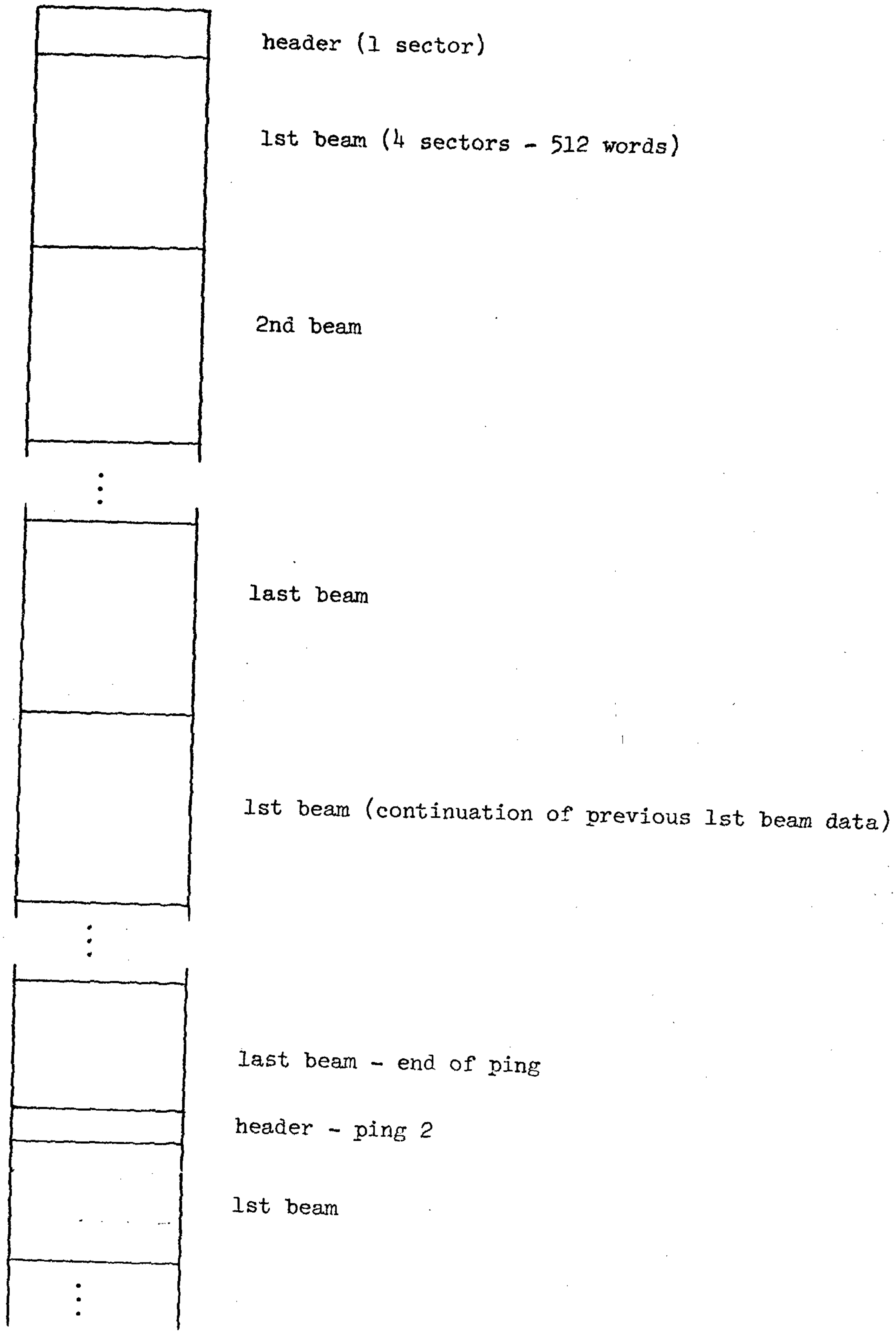
As received by the computer, each data word occupies the most significant 8 bits of the 16-bit word. Negative numbers and represented by their two's complement. Synchronization is achieved by output of a 1 to the Buoy Data interface card output register. When this is done the next input word will be word 1 of frame 1. Input is then continuous at 10667 words per second until stopped.

Figure 4. Buoy Data Input Format



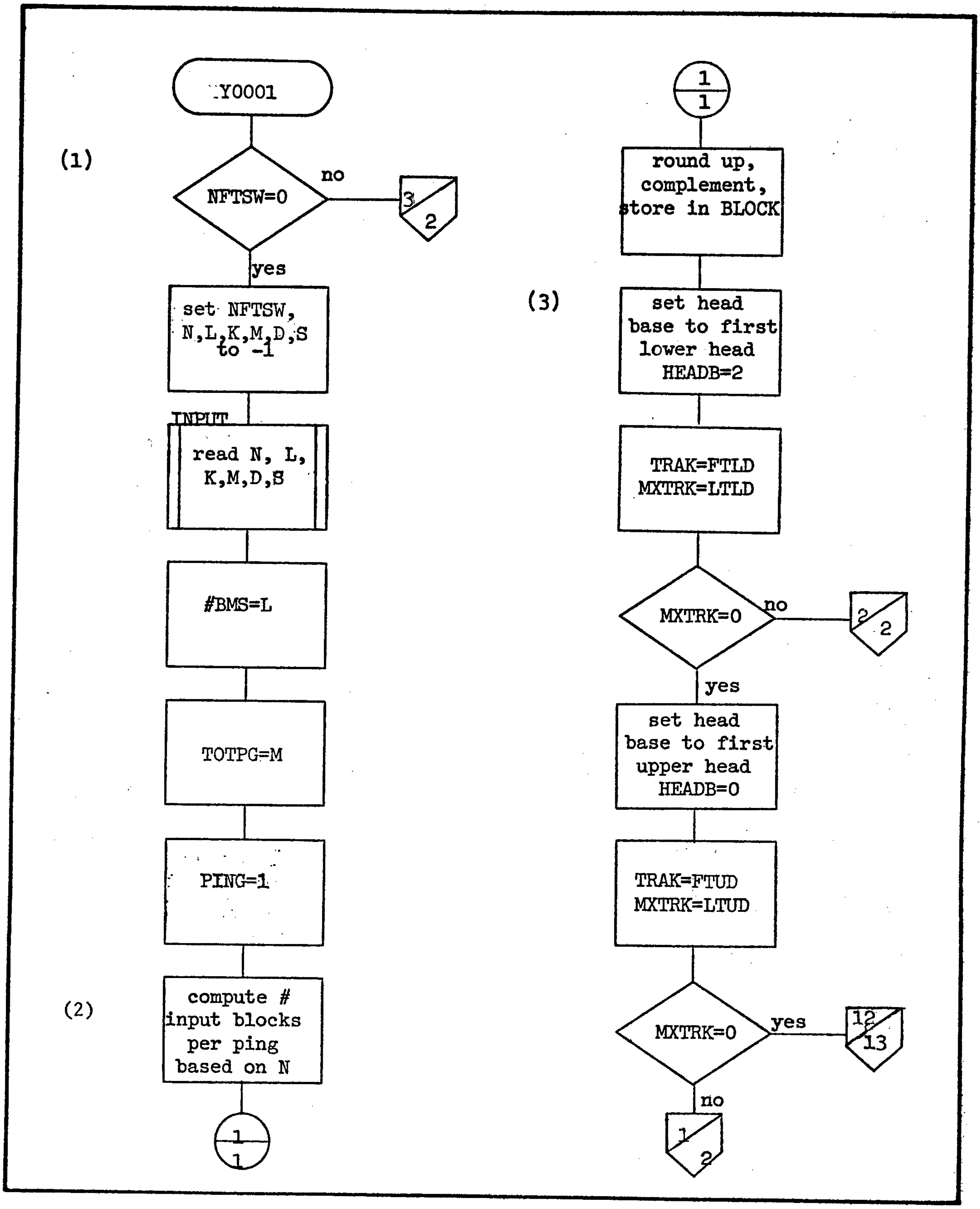

NOTE: All inputs are inverted (one's complement)

Figure 5. Time Code Generator Input Format

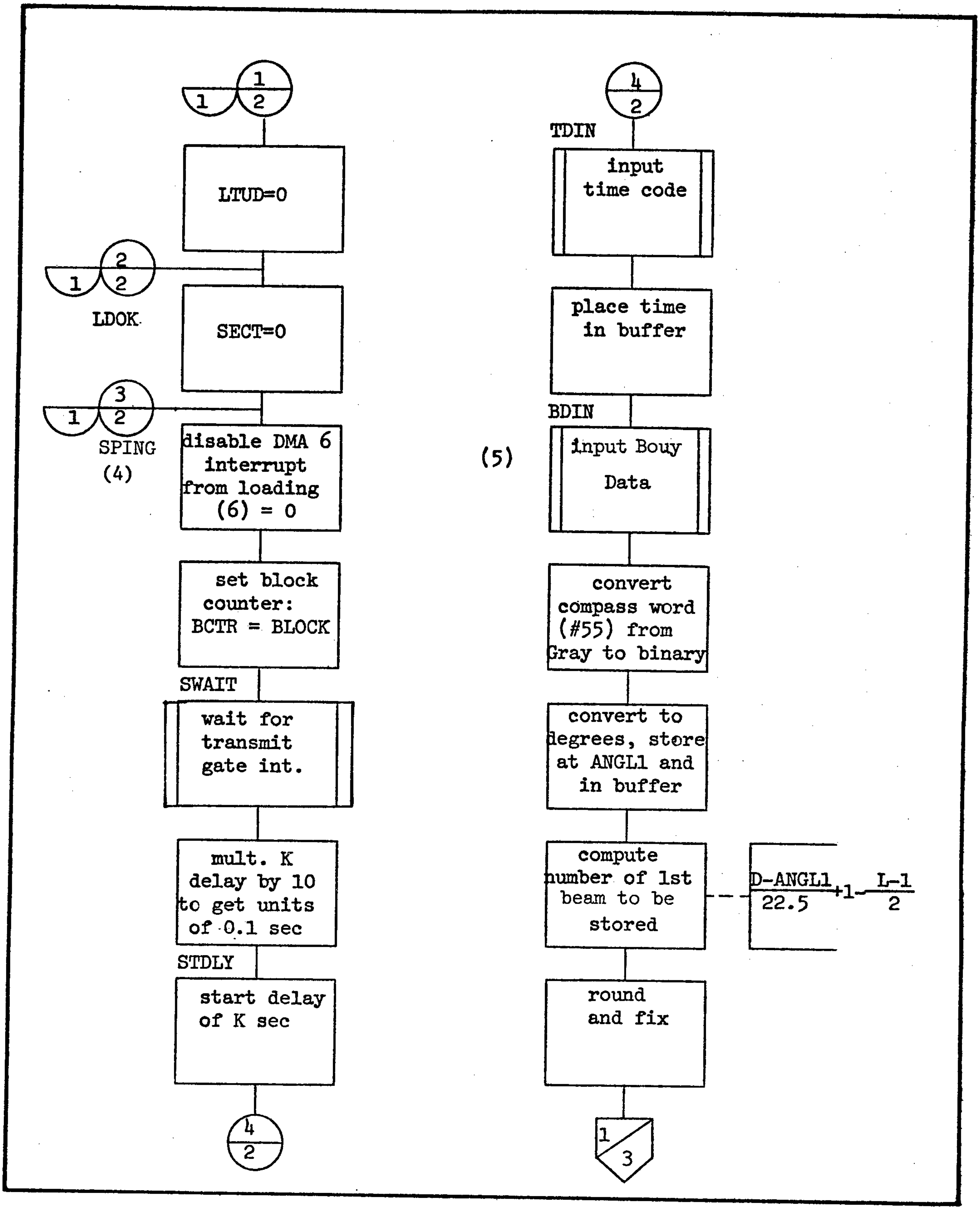



Actual amount of data recorded is dependent upon number of beams, pings, and recording time. Track boundaries occur every 48 128-word sectors.

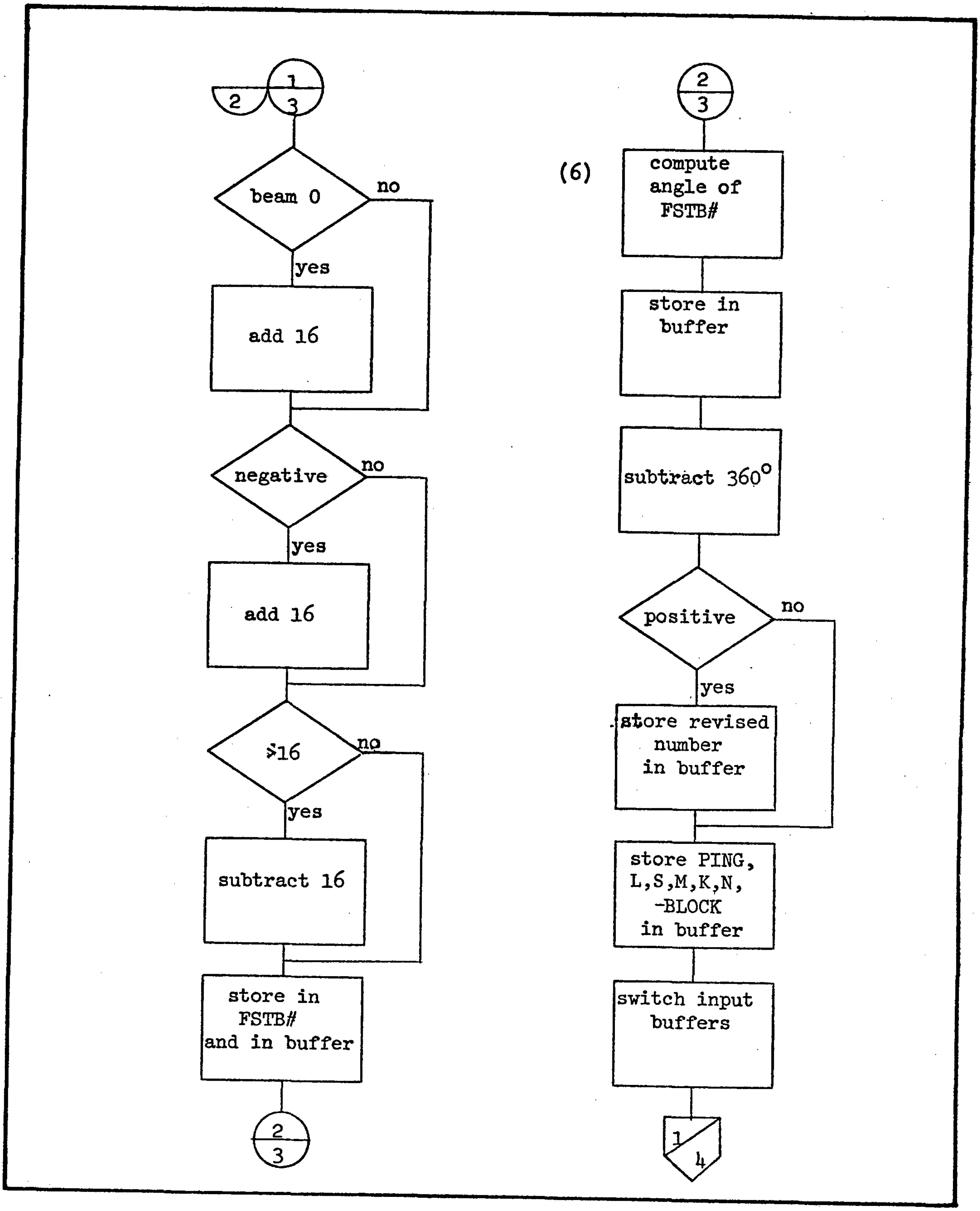
Figure 6. Disc Data Format

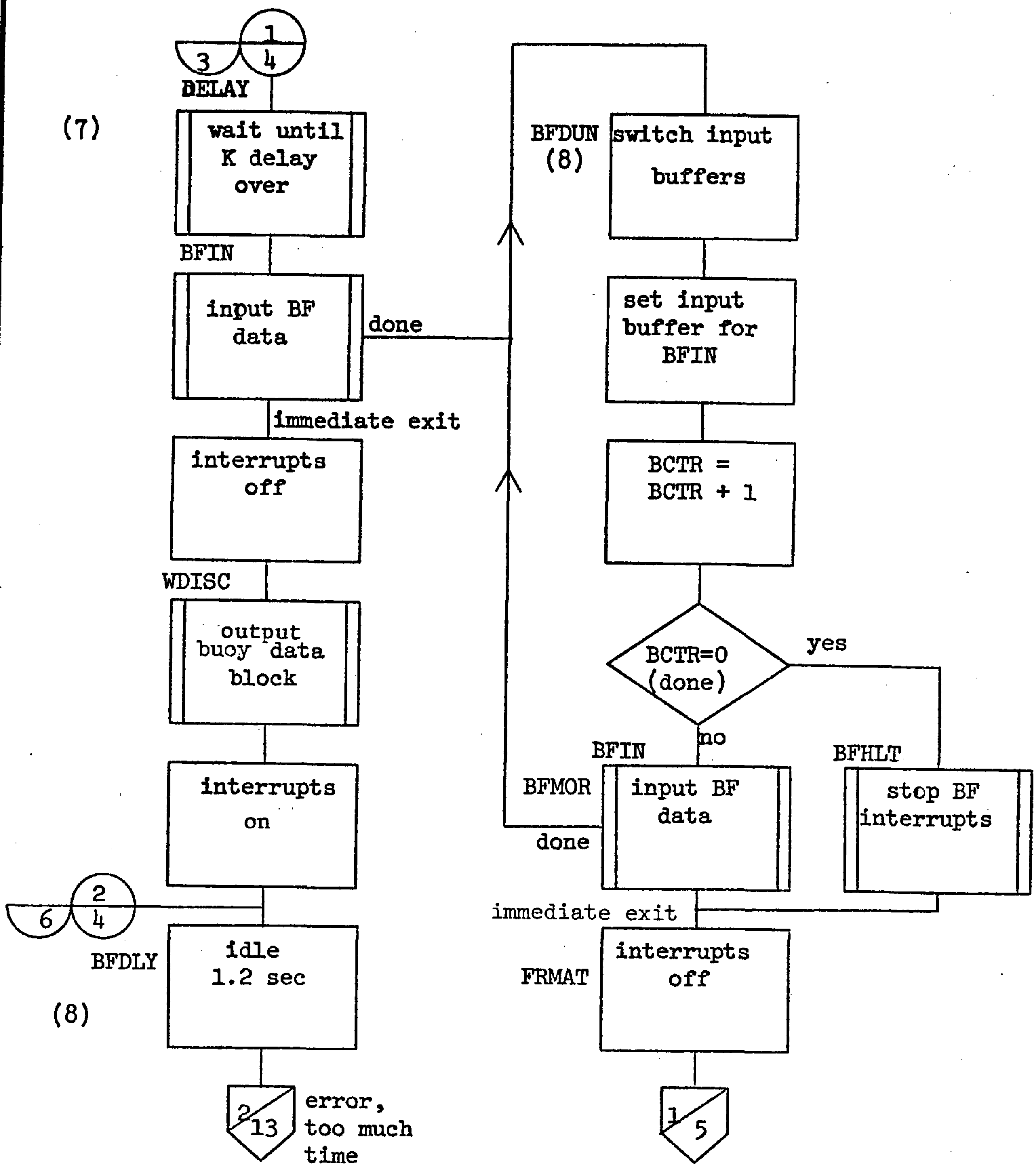


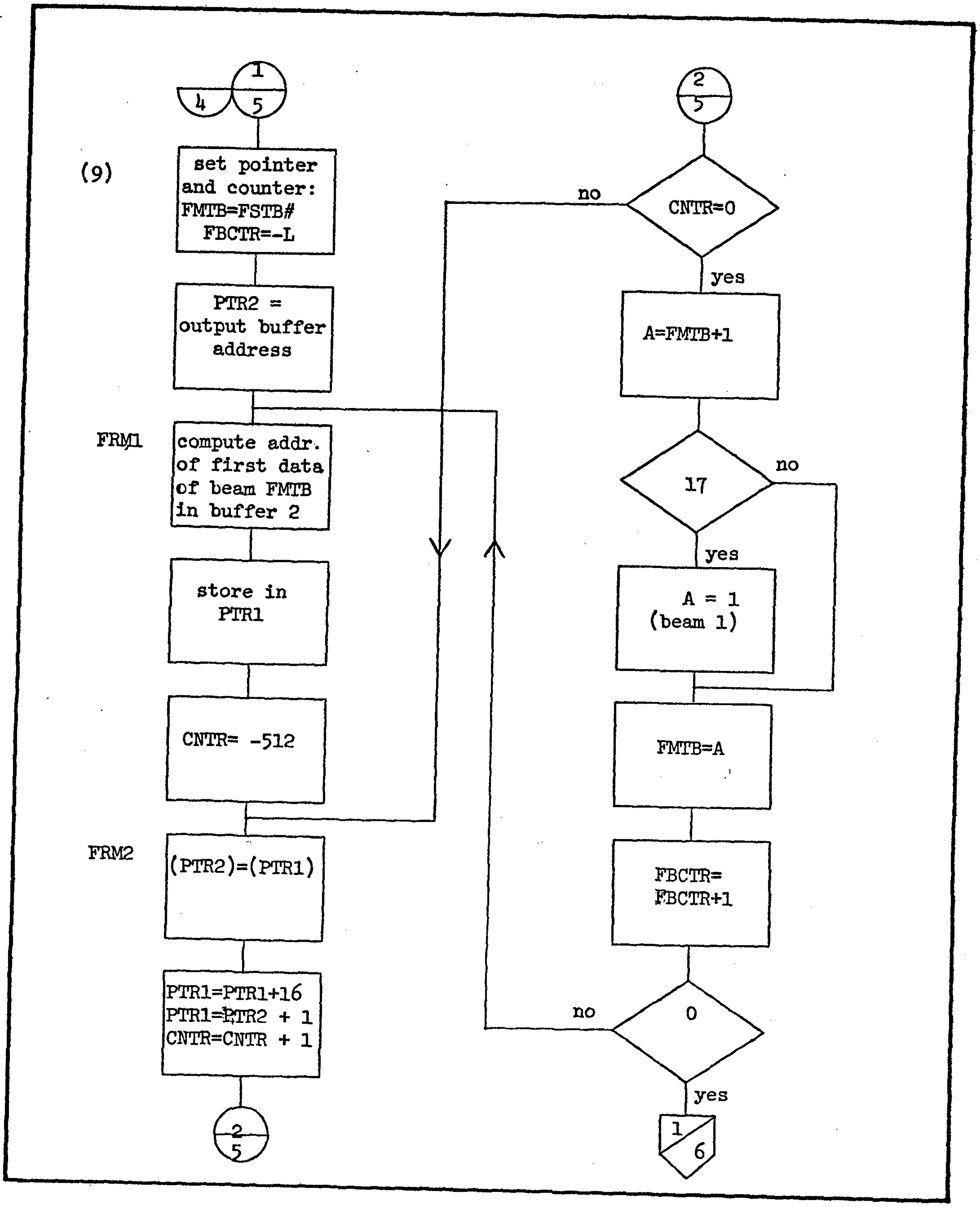
YO001
(1)
NFTSW=0
no
3/2
yes
set NFTSW, N,L,K,M,D,S to -1
INPUT
read N, L, K,M,D,S
#BMS=L
TOTPG=M
PING=1
(2)
compute # input blocks per ping based on N
1/1
1/1
round up, complement, store in BLOCK
(3)
set head base to first lower head HEADB=2
TRAK=FTLD
MXTRK=LTLD
MXTRK=0
no
2/2
yes
set head base to first upper head HEADB=0
TRAK=FTUD
MXTRK=LTUD
MXTRK=0
yes
12/13
no
1/2

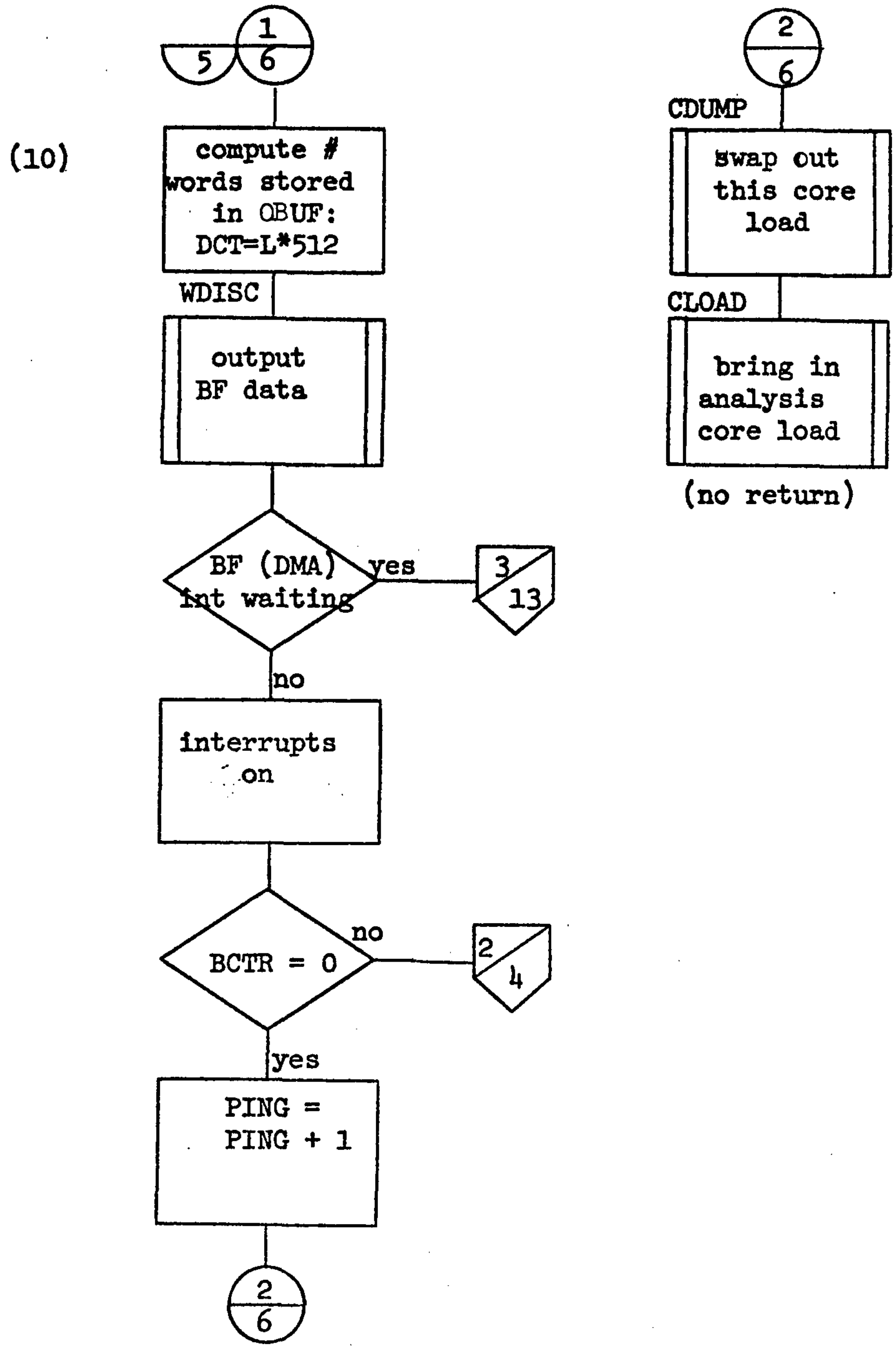


1
1 2
LTUD=0
2
1 2
LDOK
SECT=0
3
1 2
SPING
(4)
disable DMA 6 interrupt from loading (6) = 0
set block counter: BCTR = BLOCK
SWAIT
wait for transmit gate int.
mult. K delay by 10 to get units of 0.1 sec
STDLY
start delay of K sec
4
2
(5)
4
2
TDIN
input time code
place time in buffer
BDIN
input Bouy Data
convert compass word (#55) from Gray to binary
convert to degrees, store at ANGL1 and in buffer
compute number of 1st beam to be stored
D-ANGL1 / 22.5 +1- L-1 / 2
round and fix
1
3

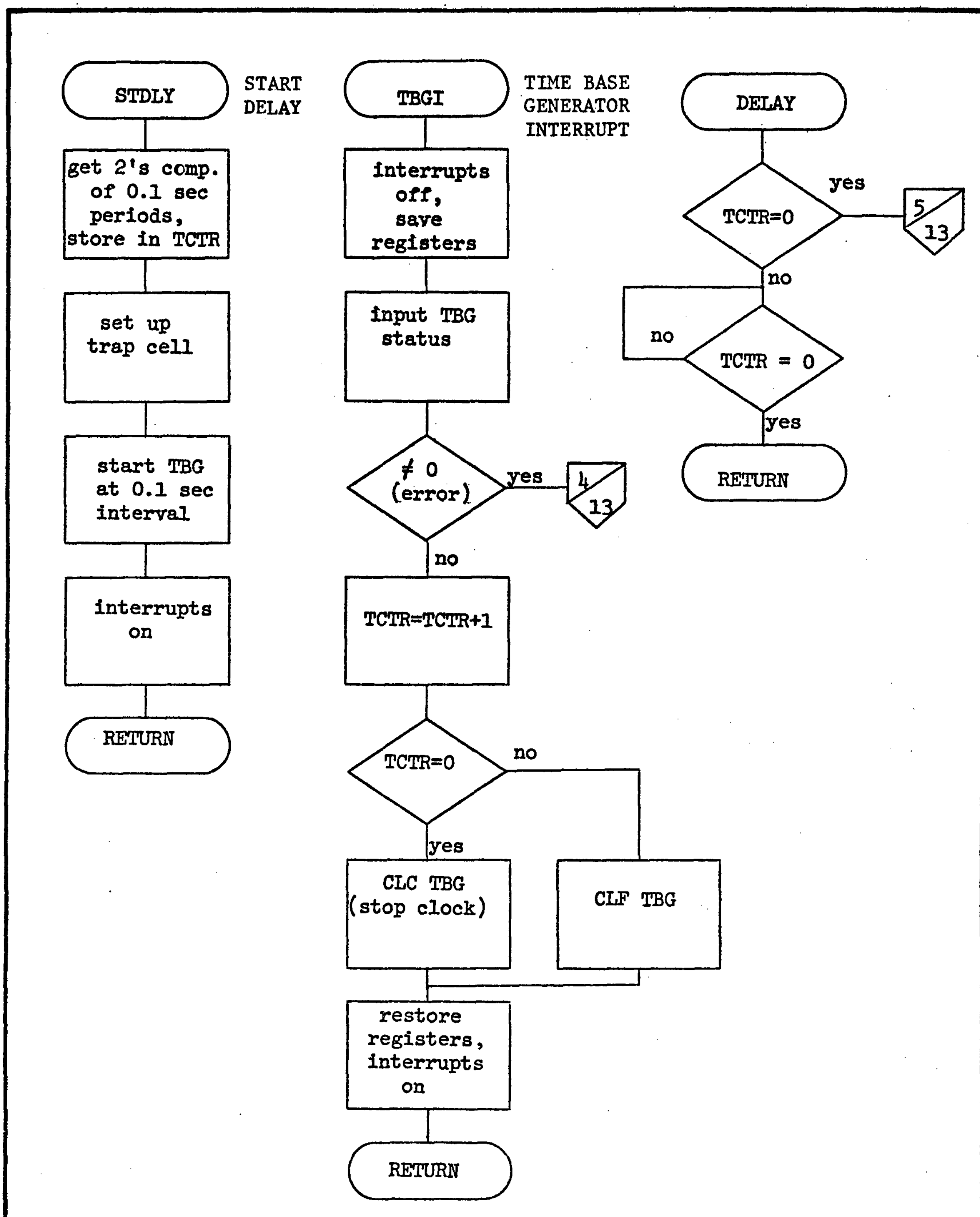

1
3
2
beam 0
no
yes
add 16
negative
no
yes
add 16
>16
no
yes
subtract 16
store in FSTB# and in buffer
2
3
2
3
(6)
compute angle of FSTB#
store in buffer
subtract 360°
positive
no
yes
store revised number in buffer
store PING, L,S,M,K,N, -BLOCK in buffer
switch input buffers
1
4

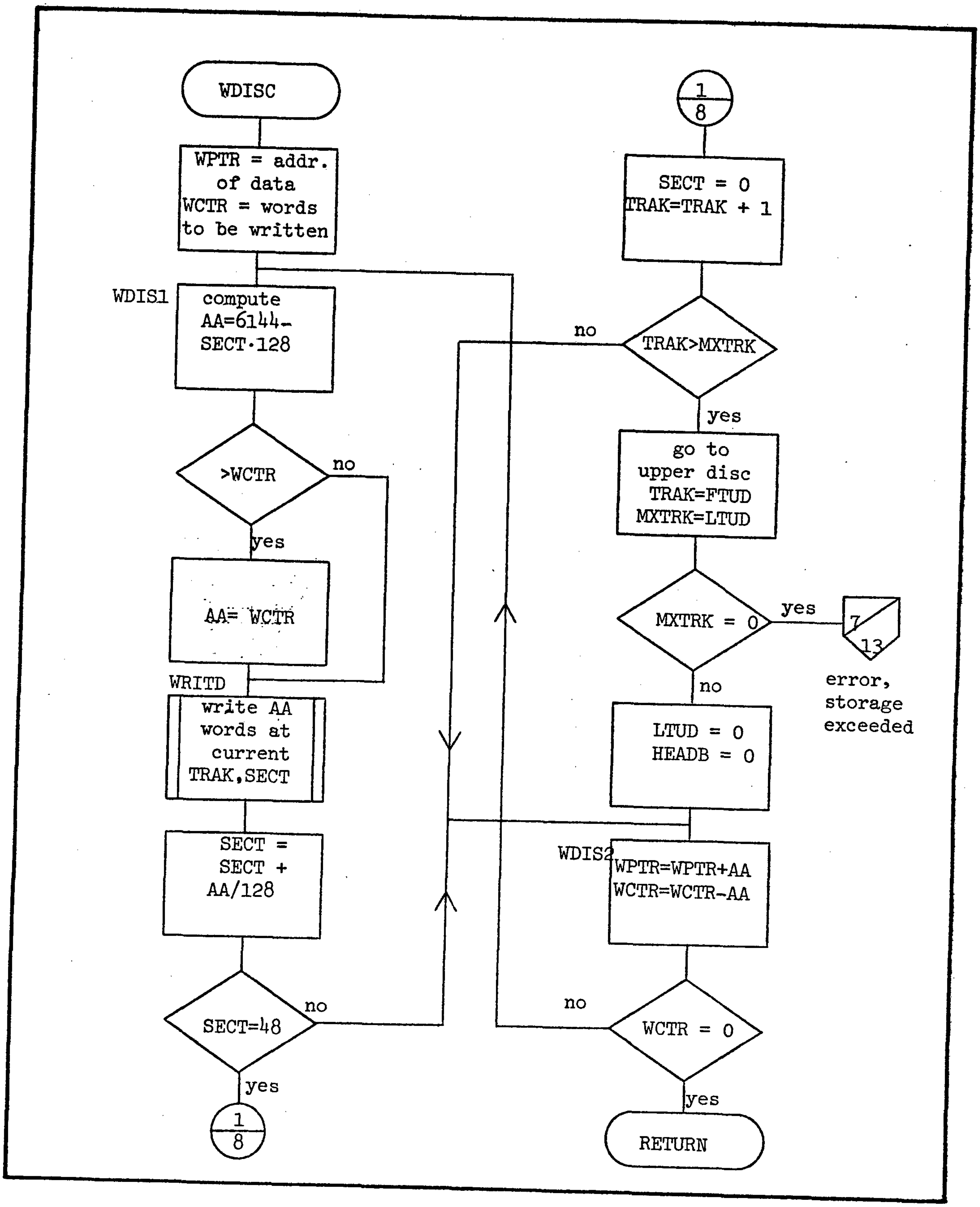

1
4
3
DELAY
(7)
wait until K delay over
BFIN
input BF data
done
immediate exit
interrupts off
WDISC
output buoy data block
interrupts on
2
4
6
BFDLY
idle 1.2 sec
(8)
2
13
error, too much time
BFDUN
(8)
switch input buffers
set input buffer for BFIN
BCTR = BCTR + 1
BCTR=0 (done)
yes
no
BFIN
BFMOR
input BF data
done
BFHLT
stop BF interrupts
immediate exit
FRMAT
interrupts off
1
5


1
5
4
(9)
set pointer and counter: FMTB=FSTB# FBCTR=-L
PTR2 = output buffer address
FRM1
compute addr. of first data of beam FMTB in buffer 2
store in PTR1
CNTR= -512
FRM2
(PTR2)=(PTR1)
PTR1=PTR1+16
PTR1=PTR2 + 1
CNTR=CNTR + 1
2
5
CNTR=0
no
yes
A=FMTB+1
17
no
yes
A = 1 (beam 1)
FMTB=A
FBCTR= FBCTR+1
0
no
yes
1
6

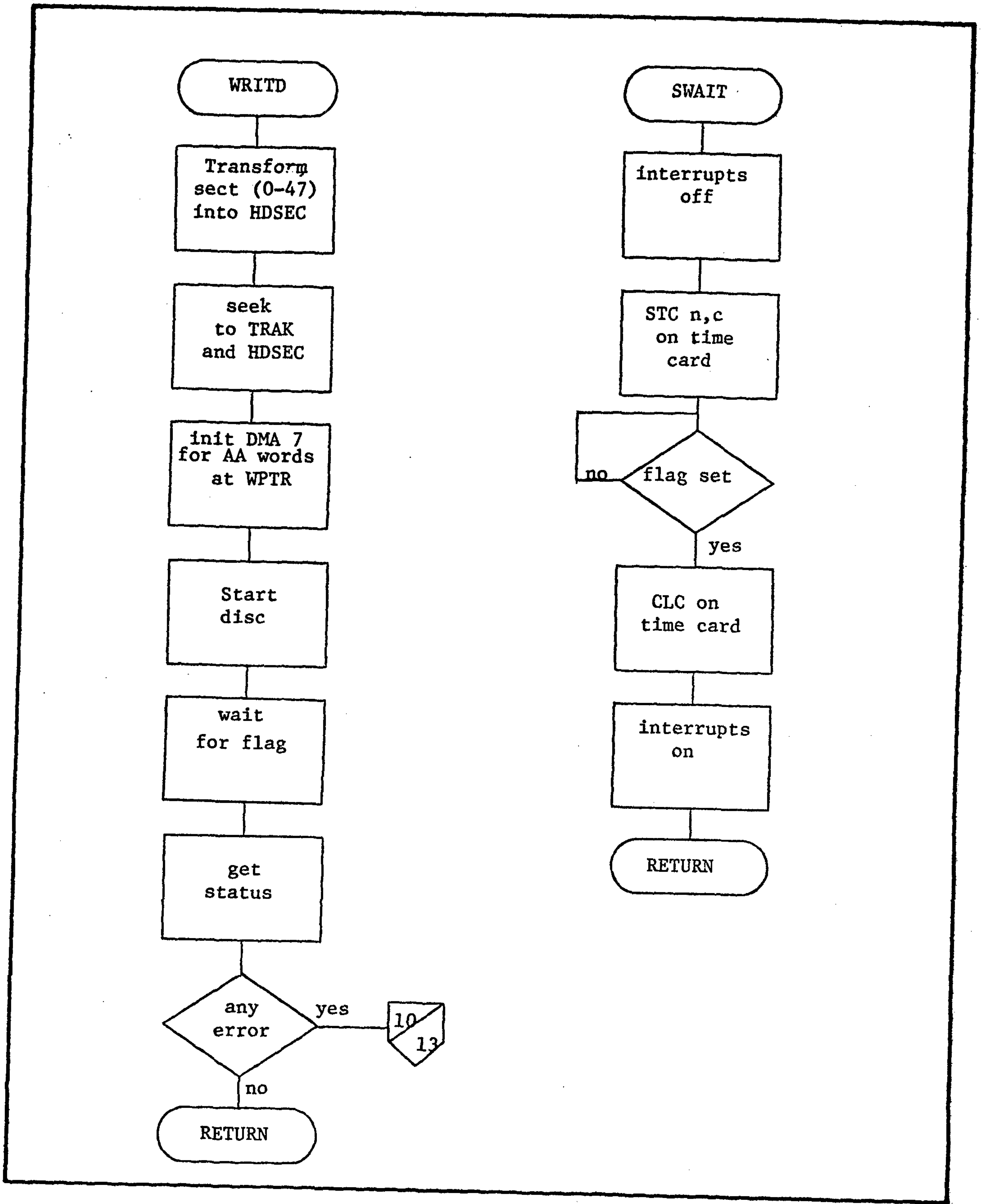


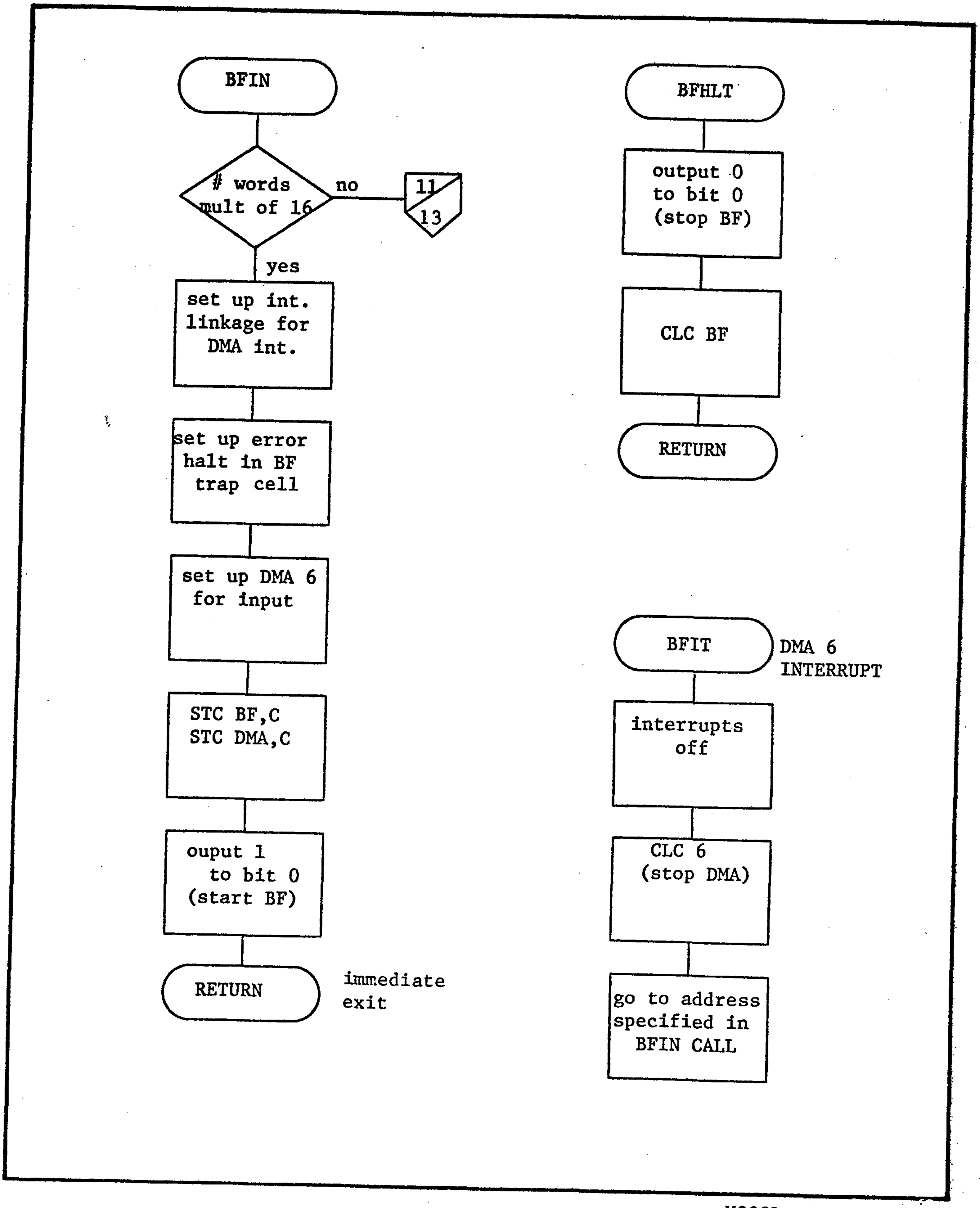
1/6
5
(10)
compute # words stored in OBUF: DCT=L*512
WDISC
output BF data
BF (DMA) int waiting
yes
3/13
no
interrupts on
BCTR = 0
no
2/4
yes
PING = PING + 1
2/6
2/6
CDUMP
swap out this core load
CLOAD
bring in analysis core load
(no return)

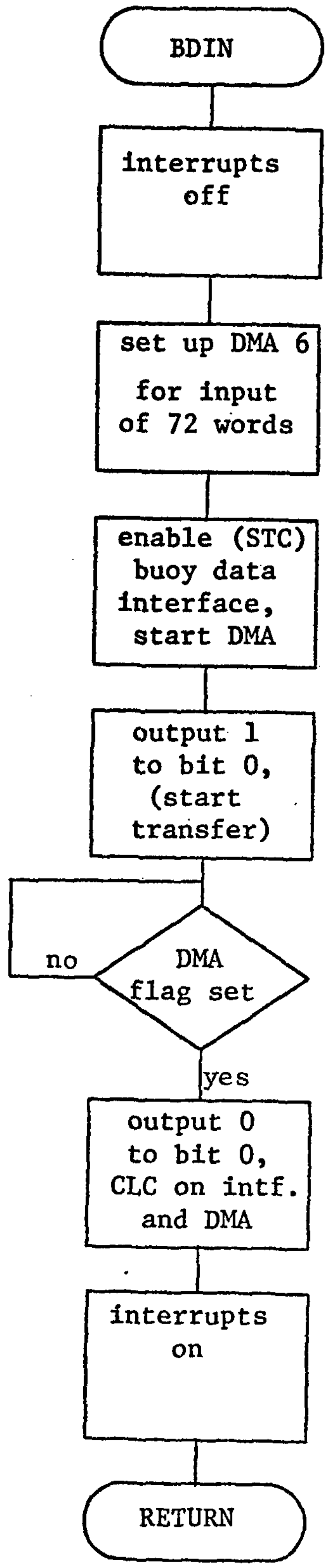

STDLY
START DELAY
get 2's comp. of 0.1 sec periods, store in TCTR
set up trap cell
start TBG at 0.1 sec interval
interrupts on
RETURN
TBGI
TIME BASE GENERATOR INTERRUPT
interrupts off, save registers
input TBG status
≠ 0 (error)
yes
4/13
no
TCTR=TCTR+1
TCTR=0
no
yes
CLC TBG (stop clock)
CLF TBG
restore registers, interrupts on
RETURN
DELAY
TCTR=0
yes
5/13
no
no
TCTR = 0
yes
RETURN



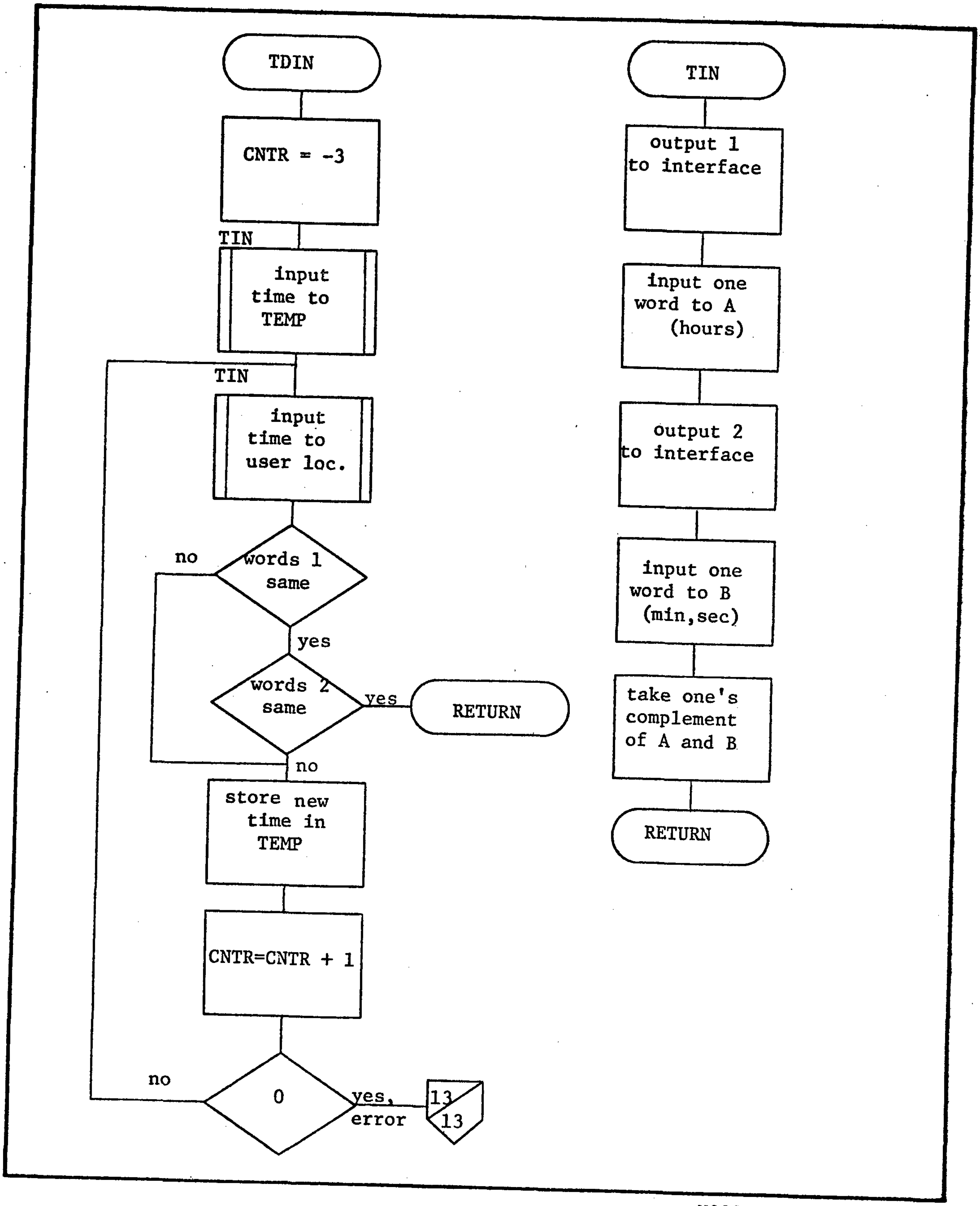
WDISC
WPTR = addr. of data
WCTR = words to be written
WDIS1
compute AA=6144-SECT·128
>WCTR
no
yes
AA= WCTR
WRITD
write AA words at current TRAK,SECT
SECT = SECT + AA/128
SECT=48
no
yes
1
8
1
8
SECT = 0
TRAK=TRAK + 1
TRAK>MXTRK
no
yes
go to upper disc
TRAK=FTUD
MXTRK=LTUD
MXTRK = 0
yes
7
13
error, storage exceeded
no
LTUD = 0
HEADB = 0
WDIS2
WPTR=WPTR+AA
WCTR=WCTR-AA
WCTR = 0
no
yes
RETURN

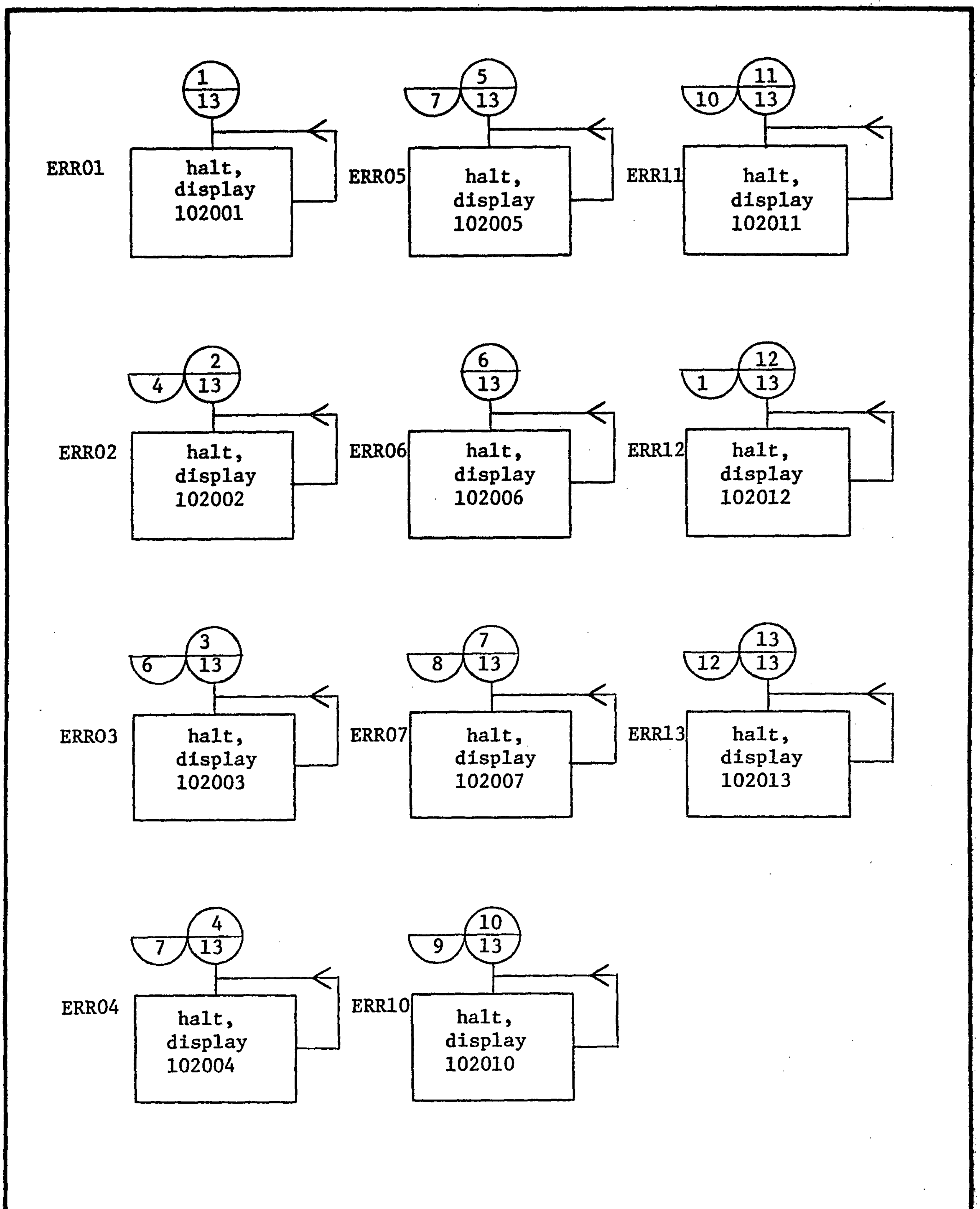

WRITD
Transform sect (0-47) into HDSEC
seek to TRAK and HDSEC
init DMA 7 for AA words at WPTR
Start disc
wait for flag
get status
any error
yes
10
13
no
RETURN
SWAIT
interrupts off
STC n,c on time card
no
flag set
yes
CLC on time card
interrupts on
RETURN


BFIN
# words mult of 16
no
11
13
yes
set up int. linkage for DMA int.
set up error halt in BF trap cell
set up DMA 6 for input
STC BF,C
STC DMA,C
ouput 1 to bit 0 (start BF)
RETURN
immediate exit
BFHLT
output 0 to bit 0 (stop BF)
CLC BF
RETURN
BFIT
DMA 6 INTERRUPT
interrupts off
CLC 6 (stop DMA)
go to address specified in BFIN CALL



BDIN
interrupts off
set up DMA 6 for input of 72 words
enable (STC) buoy data interface, start DMA
output 1 to bit 0, (start transfer)
DMA flag set
no
yes
output 0 to bit 0, CLC on intf. and DMA
interrupts on
RETURN


TDIN
CNTR = -3
TIN
input time to TEMP
TIN
input time to user loc.
no
words 1 same
yes
words 2 same
yes
RETURN
no
store new time in TEMP
CNTR=CNTR + 1
no
0
yes, error
13
13
TIN
output 1 to interface
input one word to A (hours)
output 2 to interface
input one word to B (min,sec)
take one's complement of A and B
RETURN


1 13
ERR01
halt, display 102001
7 5 13
ERR05
halt, display 102005
10 11 13
ERR11
halt, display 102011
4 2 13
ERR02
halt, display 102002
6 13
ERR06
halt, display 102006
1 12 13
ERR12
halt, display 102012
6 3 13
ERR03
halt, display 102003
8 7 13
ERR07
halt, display 102007
12 13 13
ERR13
halt, display 102013
7 4 13
ERR04
halt, display 102004
9 10 13
ERR10
halt, display 102010

```
                 ASMB,R,F,L
      00000             NAM Y0001
                        ENT Y0001,BFIT,TBGI
                        EXT .ENTR,INPUT,LKTBG,LINK6,CDUMP,CLOAD
                        EXT FTLD,LTLD,FTUD,LTUD
*
 ******************************************************************
 *                                                                *
 *  TITLE:  OLAAS RAW DATA COLLECTION SUBROUTINE                  *
 *                                                                *
 *  MODULE: Y0001          SOURCE: Y0001       VERSION:  X00      *
 *                         OBJECT: #Y0001                         *
 *                                                                *
 *  AUTHOR: R. TASHJIAN              DATE: 1976                   *
 *                                                                *
 *  MODULE FUNCTION:  COLLECT BEAMFORMER RAW DATA AND STORE        *
 *  THAT DATA ON THE OLAAS FOURIER SYSTEM DISC.                   *
 *                                                                *
 *  CALLING SEQUENCE:     CALLED BY - Y1002 (ANALYSIS PGM.)       *
 *                                                                *
 *     CALL CLOAD (15B,2)  (FOR CORELOAD 2)                       *
 *            OR                                                  *
 *     CALL CLOAD (31B,2)  (FOR CORELOAD 4)                       *
```

```
*
****************************************************************          *
*
00000 000000  Y0001 NOP
00001 106700        CLC 0        RESET HARDWARE
00002 063115R       LDA NFTSW    GET NOT FIRST TIME SWITCH
00003 002002        SZA
00004 026067R       JMP SPING    NOT 1ST, COLLECT NEXT PING
*
00005 003400        CCA
00006 073115R       STA NFTSW    SET NOT FIRST TIME SWITCH
00007 073073R       STA N        SET ALL INPUT PARAMETERS
00010 073075R       STA K          TO ILLEGAL VALUES (-1)
00011 073074R       STA L
00012 073076R       STA M
00013 073077R       STA D
00014 073100R       STA S
*
00015 016002X       JSB INPUT    INPUT PARAMETERS FROM TTY
00016 000025R       DEF *+7
00017 001073R       DEF N        PULSE TIME (AFTER K)
00020 001075R       DEF K        DELAY BEFORE RECORD
00021 001074R       DEF L        # BEAMS TO RECORD
00022 001076R       DEF M        TOTAL # PINGS
00023 001077R       DEF D        DESIRED DIRECTION OF CENTER BEAM
00024 001100R       DEF S        SOURCE (1-ON LINE, 0-RECORDER)
*
00025 063074R       LDA L
00026 073101R       STA #BMS
00027 063076R       LDA M
00030 073102R       STA TOTPG
00031 002404        CLA,INA
00032 073103R       STA PING
```

```
00033 063073R       LDA N
00034 105120        FLT
00035 105040        FMP =F1.302083 COMPUTE # BLOCKS PER PING
00036 061240R
00037 105100        FIX
00040 002004        INA             ROUND UP
00041 003004        CMA,INA
00042 073104R       STA BLOCK
*
00043 063242R       LDA =D2         INITIALIZE HEAD, TRACK, SECTOR
00044 072656R       STA HEADB       HEAD 2 IS FIRST LOWER HEAD
00045 062007X       LDA FTLD
00046 072653R       STA TRAK
00047 062010X       LDA LTLD
00050 072655R       STA MXTRK
00051 002002        SZA             TEST LTLD
00052 026065R       JMP LDOK        NON-ZERO, LOWER DISC HAS SPACE
00053 002400        CLA             ZERO, SET UP UPPER DISC
00054 072656R       STA HEADB       HEAD 0 IS FIRST UPPER HEAD
00055 062011X       LDA FTUD
00056 072653R       STA TRAK
00057 062012X       LDA LTUD
00060 072655R       STA MXTRK
00061 002003        SZA,RSS
00062 027063R       JMP ERR12       ERROR, NO DISC SPACE ALLOCATED
00063 002400        CLA
00064 072012X       STA LTUD        LTUD=0 INDICATES NOW USING TOP
```

```
*
00065 002400  LDOK  CLA
00066 072654R       STA SECT
*
00067 002400  SPING CLA           DISABLE INT. FROM CORE LOADING
00070 070006        STA 6
*
00071 063104R       LDA BLOCK
00072 073105R       STA BCTR
*
00073 017031R       JSB SWAIT     WAIT FOR S INTERRUPT
*
00074 063075R       LDA K
00075 043243R       ADA =D-1      SUBTRACT 1 FROM DELAY
00076 100200        MPY =D10
00077 061244R
00100 073106R       STA TEMP
00101 016377R       JSB STDLY     START DELAY OF K-1 SECONDS
00102 000104R       DEF *+2
00103 001106R       DEF TEMP
*
00104 016444R       JSB DELAY     WAIT UNTIL K DELAY IS OVER
*
00105 063244R       LDA =D10
00106 073106R       STA TEMP
00107 016377R       JSB STDLY     START DELAY OF 1 SECOND
00110 000112R       DEF *+2
00111 001106R       DEF TEMP
*
00112 016772R       JSB TDIN      GET TIME FROM TIME INTF
```

```
00113 000115R        DEF *+2
00114 001240R        DEF OBUF       ADDR IN DISC BUFFER FOR TIME
*
00115 016736R        JSB BDIN       INPUT BUOY DATA (72 WORDS)
00116 000120R        DEF *+2
00117 001130R        DEF DTA72
00120 063216R        LDA DTA72+54 GET COMPASS WORD
00121 013245R        AND =B177400
*
00122 070001         STA B          CONVERT GRAY TO BINARY
00123 063246R        LDA =D-8
00124 073110R        STA CNTR
00125 002400         CLA
00126 002011   GB1   SLA,RSS        A ODD?
00127 026133R        JMP EVEN       NO
00130 006021         SSB,RSS        B NEGATIVE?
00131 033247R        IOR =B100000 NO
00132 026135R        JMP GB2
00133 006020   EVEN  SSB            B NEGATIVE?
00134 033247R        IOR =B100000 YES
00135 001200   GB2   RAL
00136 005200         RBL
00137 037110R        ISZ CNTR       DONE
00140 026126R        JMP GB1        NO, LOOP
00141 105120         FLT            YES, RESULT IN A
*
00142 105040         FMP =F1.40625 CONVERT TO DEGREES
00143 061250R
00144 105000         FAD =F0.5
00145 061252R
00146 105100         FIX
```

```
0141  00147 073111R      STA ANGL1     STORE: THIS IS HEADING OF BEAM 1
0142  00150 073242R      STA OBUF+2
0143*
0144  00151 063074R      LDA L         COMPUTE # OF 1ST BEAM TO
0145  00152 043243R      ADA =D-1        BE STORED
0146  00153 105120       FLT
0147  00154 105060       FDV =F2.0     (L-1)/2
      00155 061254R
0148  00156 104400       DST TEMP
      00157 001106R
0149  00160 063111R      LDA ANGL1
0150  00161 003004       CMA,INA
0151  00162 043077R      ADA D
0152  00163 002020       SSA           NEGATIVE?
0153  00164 043256R      ADA =D360     YES, ADD 1 REV
0154  00165 105120       FLT
0155  00166 105060       FDV =F22.5
      00167 061257R
0156  00170 105000       FAD =F1.0     (D-ANGL1)/22.5 + 1
      00171 061261R
0157  00172 105020       FSB TEMP      -(L-1)/2
      00173 001106R
0158  00174 105000       FAD =F0.5     ROUND
      00175 061252R
0159  00176 105100       FIX           THIS IS FIRST BEAM NUMBER
```

```
0160  00177 002003     SZA,RSS      BEAM ZERO?
0161  00200 043263R    ADA =D16     YES, MAKE POSITIVE
0162  00201 002020     SSA          NEGATIVE?
0163  00202 043263R    ADA =D16     YES, MAKE POSITIVE
0164  00203 043264R    ADA =D-17
0165  00204 002021     SSA,RSS      >16 ?
0166  00205 043265R    ADA =D-16    YES, SUBTRACT 1 REV
0167  00206 043266R    ADA =D17     RESTORE BEAM NUMBER
0168  00207 073112R    STA FSTB#    FIRST BEAM TO BE STORED
0169  00210 073244R    STA OBUF+4   STORE IN BUFFER
0170*
0171  00211 043243R    ADA =D-1     GET BEAM NUMBER DIFFERENCE
0172  00212 100200     MPY =D45     X 22.5
      00213 061267R
0173  00214 043111R    ADA ANGL1    + ANGL1
0174  00215 043111R    ADA ANGL1      (X2 DUE TO SCALE FACTOR)
0175  00216 002004     INA          ROUND
0176  00217 000065     CLE,ERA      RESCALE
0177  00220 073245R    STA OBUF+5   POSITIVE ANGLE OF FSTB#
0178  00221 043270R    ADA =D-360
0179  00222 002021     SSA,RSS      >360 ?
0180  00223 073245R    STA OBUF+5   YES, STORE 1 REV LESS
0181*
0182  00224 063103R    LDA PING
0183  00225 073246R    STA OBUF+6   PING # TO BUFFER
0184  00226 063074R    LDA L
0185  00227 073247R    STA OBUF+7   # BEAMS
0186  00230 063100R    LDA S
0187  00231 073250R    STA OBUF+8   SOURCE
0188  00232 063076R    LDA M
0189  00233 073251R    STA OBUF+9   # PINGS
0190  00234 063075R    LDA K
0191  00235 073252R    STA OBUF+10  DELAY
0192  00236 063073R    LDA N
0193  00237 073253R    STA OBUF+11  TOTAL PULSE TIME
```

```
0194  00240 063104R         LDA BLOCK
0195  00241 003004          CMA,INA
0196  00242 073254R         STA OBUF+12   # BLOCKS PER PING
0197*
0198  00243 063126R         LDA BUF1A     SWITCH BUFFERS
0199  00244 067127R         LDB BUF2A
0200  00245 073127R         STA BUF2A
0201  00246 077126R         STB BUF1A
0202*
0203  00247 016444R         JSB DELAY     WAIT UNTIL K DELAY OVER
0204*
0205  00250 063126R         LDA BUF1A
0206  00251 072254R         STA INBUF
0207  00252 016666R         JSB BFIN      START INPUT OF BF DATA
0208  00253 000257R         DEF *+4
0209  00254 000000   INBUF  NOP           INPUT BUFFER ADDRESS
0210  00255 020000          DEC 8192
0211  00256 000275R         DEF BFDUN     COMPLETION ADDRESS
0212*
0213  00257 103100          CLF 0         INTERRUPTS OFF
0214*
```



```
0215  00260 016462R         JSB WDISC     OUTPUT BUOY DATA BLOCK
0216  00261 000264R         DEF *+3
0217  00262 001240R         DEF OBUF
0218  00263 000200          DEC 128
```

```
*
00264 102100         STF 0        INTERRUPTS ON
00265 063271R BFDLY  LDA =D-3     ALLOW 1.2 SEC FOR INTERRUPT
00266 073110R        STA CNTR
00267 006400  DLY1   CLB          0.3211264 SEC PER LOOP
00270 034001  DLY2   ISZ B
00271 026270R        JMP DLY2
00272 037110R        ISZ CNTR
00273 026267R        JMP DLY1
00274 027043R        JMP ERR02    NO BF DONE INTERRUPT, ERROR
*
00275 063126R BFDUN  LDA BUF1A    SWITCH BUFFERS
00276 067127R        LDB BUF2A
00277 073127R        STA BUF2A
00300 077126R        STB BUF1A
00301 076310R        STB IBUFA    SET INPUT BUFFER FOR BFIN
00302 037105R        ISZ BCTR     INCR BCTR  =0?
00303 026306R        JMP BFMOR    NO
00304 016723R        JSB BFHLT    YES, STOP BF DATA
00305 026313R        JMP FRMAT    GO FORMAT FINAL DATA
00306 016666R BFMOR  JSB BFIN     START INPUT OF BF DATA
00307 000313R        DEF *+4
00310 000000  IBUFA  NOP
00311 020000         DEC 8192
00312 000275R        DEF BFDUN    COMPLETION ADDRESS
*
00313 103100  FRMAT  CLF 0        INTERRUPTS OFF
00314 063112R        LDA FSTB#    SELECT AND FORMAT DATA
00315 073113R        STA FMTB
00316 063074R        LDA L
00317 003004         CMA,INA
00320 073114R        STA FBCTR
00321 063125R        LDA OBUFA
00322 073117R        STA PTR2
00323 063127R FRM1   LDA BUF2A
```

```
0254  00324 043113R       ADA FMTB
0255  00325 043243R       ADA =D-1
0256  00326 073116R       STA PTR1
0257  00327 063272R       LDA =D-512
0258  00330 073110R       STA CNTR
0259  00331 163116R FRM2  LDA PTR1,I
0260  00332 173117R       STA PTR2,I
0261  00333 063116R       LDA PTR1
0262  00334 043263R       ADA =D16
0263  00335 073116R       STA PTR1
0264  00336 037117R       ISZ PTR2
0265  00337 037110R       ISZ CNTR
0266  00340 026331R       JMP FRM2
0267  00341 063113R       LDA FMTB
0268  00342 002004        INA            INCREMENT BEAM
0269  00343 053266R       CPA =D17       17?
0270  00344 002404        CLA,INA        YES, SET TO 1
```

PAGE 0007 #01  OLAAS DATA COLLECTION PROGRAM

```
0271  00345 073113R       STA FMTB
0272  00346 037114R       ISZ FBCTR      ALL REQ'D BEAMS DONE?
0273  00347 026323R       JMP FRM1       NO
0274*
0275  00350 063074R       LDA L          YES, COMPUTE # WORDS JUST
0276  00351 100200        MPY =D512        STORED IN OBUF
      00352 061273R
0277  00353 072357R       STA DCT
```

```
*
      00354 016462R        JSB WDISC    WRITE BF DATA ON DISC
      00355 000360R        DEF *+3
      00356 001240R        DEF OBUF
      00357 000000  DCT    NOP          (DISC WORD COUNT)
*
      00360 102206         SFC 6        BF COMPL. INT. WAITING?
      00361 027045R        JMP ERR03    YES, ERROR
      00362 102100         STF 0        INTERRUPTS ON
*
      00363 063105R        LDA BCTR
      00364 002002         SZA
      00365 026265R        JMP BFDLY
      00366 037103R        ISZ PING
*           LDA PING
*           CMA,INA
*           ADA TOTPG    TOTPG - PING
*           SSA,RSS      PING GT TOTPG?
*           JMP SPING    NO
*
      00367 016005X        JSB CDUMP    SWAP OUT THIS CORE LOAD
      00370 000372R        DEF *+2
      00371 001124R        DEF CYLCO
      00372 016006X        JSB CLOAD    BRING IN NEXT CORE LOAD
      00373 000376R        DEF *+3
      00374 001123R        DEF CYLAN
      00375 001121R        DEF XFER
*
* DELAY SUBROUTINE
*
      00376 000000  DTIM   NOP
      00377 000000  STDLY  NOP
      00400 016001X        JSB .ENTR
      00401 000376R        DEF DTIM
      00402 162376R        LDA DTIM,I   SET COUNT
```

```
0313  00403 003004          CMA,INA
0314  00404 072454R         STA TCTR
0315  00405 062003X         LDA LKTBG    SET UP TRAP CELL
0316  00406 070013          STA TBG
0317  00407 063274R         LDA =D3      SET TBG TO 100 MS INTERVAL
0318  00410 102613          OTA TBG
0319  00411 103713          STC TBG,C
0320  00412 102100          STF 0        INTERRUPTS ON
0321  00413 126377R         JMP STDLY,I  RETURN
0322*
0323  00414 000000    TBGI  NOP          TBG INTERRUPT PROCESSOR
0324  00415 103100          CLF 0        INTERRUPTS OFF
0325  00416 072455R         STA SAVEA
```

PAGE 0008 #01  OLAAS DATA COLLECTION PROGRAM

```
0326  00417 076456R         STB SAVEB
0327  00420 001520          ERA,ALS
0328  00421 102201          SOC
0329  00422 002004          INA
0330  00423 072457R         STA SAVEO
0331  00424 102513          LIA TBG      READ TBG STATUS
0332  00425 002002          SZA
0333  00426 027047R         JMP ERR04    NON-ZERO:  ERROR
0334  00427 036454R         ISZ TCTR
0335  00430 026442R         JMP TNZRO
0336  00431 106713          CLC TBG      TCTR=0, STOP CLOCK
0337  00432 062457R TBGIX   LDA SAVEO    RESTORE REGISTERS
```

```
0338  00433 103101         CLO
0339  00434 000036         SLA,ELA
0340  00435 102101         STO
0341  00436 062455R        LDA SAVEA
0342  00437 066456R        LDB SAVEB
0343  00440 102100         STF 0          INTERRUPTS ON
0344  00441 126414R        JMP TBGI,I     RETURN
0345  00442 103113  TNZRO  CLF TBG        CLEAR FLAG TO WAIT FOR NEXT INT.
0346  00443 026432R        JMP TBGIX
0347*
0348  00444 000000  DELAY  NOP
0349  00445 062454R        LDA TCTR
0350  00446 002003         SZA,RSS
0351  00447 027051R        JMP ERR05      ERROR, TIME OVER AT ENTRY
0352  00450 062454R DELA1  LDA TCTR
0353  00451 002002         SZA
0354  00452 026450R        JMP DELA1      LOOP UNTIL TCTR=0
0355  00453 126444R        JMP DELAY,I    RETURN
0356*
0357  00454 000000  TCTR   NOP
0358  00455 000000  SAVEA  NOP
0359  00456 000000  SAVEB  NOP
0360  00457 000000  SAVEO  NOP
0361*
0362* DISC SUBROUTINES
0363*
0364  00460 000000  WPTR   NOP
0365  00461 000000  WCTR   NOP
0366  00462 000000  WDISC  NOP
0367  00463 016001X        JSB .ENTR
0368  00464 000460R        DEF WPTR
0369*
0370  00465 062654R WDIS1  LDA SECT
0371  00466 100200         MPY =D128
      00467 061275R
```

```
0372   00470 003004          CMA,INA
0373   00471 043276R         ADA =D6144     AA = 6144 - SECT*128
0374   00472 073120R         STA AA         # WORDS REMAINING IN TRACK
0375   00473 066461R         LDB WCTR
0376   00474 003004          CMA,INA
0377   00475 042461R         ADA WCTR
0378   00476 002020          SSA            AA .GT. WCTR?
0379   00477 077120R         STB AA         YES, SUBSTITUTE WCTR FOR AA
0380   00500 016547R         JSB WRITD      WRITE AA WORDS
```



```
0381*
0382   00501 006400          CLB            UPDATE SECTOR
0383   00502 063120R         LDA AA         THIS ROUTINE USES AN ARTIFICIAL
0384   00503 100400          DIV =D128        SECTOR IN RANGE 0-47
       00504 061275R
0385   00505 006002          SZB
0386   00506 027053R         JMP ERR06      ERROR, LENGTH NOT MULT. OF 128
0387   00507 042654R         ADA SECT
0388   00510 072654R         STA SECT       SECT = SECT + AA/128
0389*
0390   00511 053277R         CPA =D48
0391   00512 002001          RSS
0392   00513 026535R         JMP WDIS2      SECTOR NOT 48, SKIP TRACK UPDATE
0393   00514 002400          CLA
0394   00515 072654R         STA SECT
0395   00516 036653R         ISZ TRAK
```

```
0396  00517 062653R        LDA TRAK
0397  00520 003004         CMA,INA
0398  00521 042655R        ADA MXTRK
0399  00522 002021         SSA,RSS      TRAK .GT. MXTRK?
0400  00523 026535R        JMP WDIS2    NO
0401  00524 062011X        LDA FTUD     YES, GO TO SECOND DISC
0402  00525 072653R        STA TRAK
0403  00526 062012X        LDA LTUD
0404  00527 072655R        STA MXTRK
0405  00530 002003         SZA,RSS
0406  00531 027055R        JMP ERR07    ERROR, STORAGE EXCEEDED
0407  00532 002400         CLA
0408  00533 072012X        STA LTUD     LTUD=0 INDICATES NOW USING TOP
0409  00534 072656R        STA HEADB    HEAD BASE = 1ST UPPR DISC HEAD
0410*
0411  00535 063120R WDIS2  LDA AA       UPDATE POINTER AND COUNTER
0412  00536 042460R        ADA WPTR
0413  00537 072460R        STA WPTR
0414  00540 063120R        LDA AA
0415  00541 003004         CMA,INA
0416  00542 042461R        ADA WCTR
0417  00543 072461R        STA WCTR
0418  00544 002002         SZA          WCTR=0?
0419  00545 026465R        JMP WDIS1    NO, GO DO ANOTHER WRITE
0420  00546 126462R        JMP WDISC,I  RETURN
0421*
0422  00547 000000  WRITD  NOP
0423  00550 062654R        LDA SECT     CONVERT SECT TO HEAD AND SECTOR
0424  00551 070001         STA B
0425  00552 043300R        ADA =D-24
0426  00553 002020         SSA
0427  00554 026561R        JMP FRSTH
0428  00555 070001         STA B
0429  00556 062656R        LDA HEADB
0430  00557 002004         INA
```

```
0431   00560 002001          RSS
0432   00561 062656R FRSTH   LDA HEADB
0433   00562 001727          ALF,ALF
0434   00563 030001          IOR B
0435   00564 072657R         STA HDSEC
```



```
0436   00565 016615R         JSB SEEK
0437   00566 106717          CLC DISCD
0438   00567 106720          CLC DISCC
0439   00570 063301R         LDA =B120000 SET UP DMA
0440   00571 032662R         IOR SELDD
0441   00572 102607          OTA 7
0442   00573 106703          CLC 3
0443   00574 062460R         LDA WPTR
0444   00575 102603          OTA 3
0445   00576 102703          STC 3
0446   00577 063120R         LDA AA      # WORDS TO WRITE
0447   00600 003004          CMA,INA
0448   00601 102603          OTA 3
0449   00602 102717          STC DISCD
0450   00603 102117          STF DISCD
0451   00604 103707          STC 7,C
0452   00605 062660R         LDA WRCMD
0453   00606 102620          OTA DISCC
0454   00607 103720          STC DISCC,C
0455   00610 102320          SFS DISCC   WAIT FOR COMPLETION
```

```
00611 026610R       JMP *-1
00612 106707        CLC 7       STOP DMA
00613 016637R       JSB STATS   CHECK STATUS
00614 126547R       JMP WRITD,I RETURN
*
00615 000000  SEEK  NOP
00616 106717        CLC DISCD
00617 106720        CLC DISCC
00620 062653R       LDA TRAK
00621 102617        OTA DISCD
00622 103717        STC DISCD,C
00623 062661R       LDA SKCMD
00624 102620        OTA DISCC
00625 103720        STC DISCC,C
00626 102317        SFS DISCD
00627 026626R       JMP *-1
00630 062657R       LDA HDSEC
00631 102617        OTA DISCD
00632 103717        STC DISCD,C
00633 102320        SFS DISCC
00634 026633R       JMP *-1
00635 016637R       JSB STATS
00636 126615R       JMP SEEK,I
*
00637 000000  STATS NOP
00640 106717        CLC DISCD
00641 106720        CLC DISCC
00642 002400        CLA         STATUS COMMAND
00643 102620        OTA DISCC
00644 103720        STC DISCC,C
00645 102317        SFS DISCD
00646 026645R       JMP *-1
00647 102517        LIA DISCD
00650 000010        SLA
00651 027057R       JMP ERR10   ERROR, DISC STATUS BAD
```

```
0491   00652 126637R        JMP STATS,I   RETURN
```



```
0492*
0493   00653 000000   TRAK  NOP
0494   00654 000000   SECT  NOP
0495   00655 000000   MXTRK NOP
0496   00656 000000   HEADB NOP
0497   00657 000000   HDSEC NOP
0498   00660 010000   WRCMD OCT 10000
0499   00661 030000   SKCMD OCT 30000
0500   00662 000017   SELDD DEF DISCD
0501*
0502* BEAM FORMER DRIVER
0503*
0504*
0505   00663 000000   BDATA NOP                ADDR
0506   00664 000000   BFCT  NOP                COUNT
0507   00665 000000   COMPL NOP                COMPLETION ADDRESS
0508   00666 000000   BFIN  NOP                BEAM FORMER INPUT INITIATOR
0509   00667 016001X        JSB .ENTR
0510   00670 000663R        DEF BDATA
0511   00671 062664R        LDA BFCT           GET COUNT
0512   00672 006400         CLB
0513   00673 100400         DIV =D16           VERIFY DIVISIBLE BY 16
       00674 061263R
0514   00675 006002         SZB
```

```
00676 027061R        JMP ERR11     REMAINDER NOT 0, ERROR
00677 062004X        LDA LINK6     SET UP DMA INT. LINK
00700 070006         STA 6
00701 063122R        LDA HLT30     SET UP HALT IN BF S.C.
00702 070022         STA BF
*
00703 063301R        LDA =B120000 SET UP DMA
00704 032734R        IOR SELBF
00705 102606         OTA 6
00706 106702         CLC 2
00707 062663R        LDA BDATA
00710 033247R        IOR =B100000
00711 102602         OTA 2
00712 102702         STC 2
00713 062664R        LDA BFCT
00714 003004         CMA,INA
00715 102602         OTA 2
00716 103722         STC BF,C      ENABLE BEAM FORMER INTERRUPT
00717 103706         STC 6,C       START DMA
00720 002404         CLA,INA
00721 102622         OTA BF        START BEAM FORMER
00722 126666R        JMP BFIN,I    RETURN
*
00723 000000  BFHLT  NOP           STOP BEAM FORMER
00724 002400         CLA
00725 102622         OTA BF
00726 106722         CLC BF
00727 126723R        JMP BFHLT,I   RETURN
*
00730 000000  BFIT   NOP           BEAM FORMER DMA COMPLETION PRCSSR
00731 103100         CLF 0         INTERRUPTS OFF
00732 107706         CLC 6,C       STOP DMA
```

```
0547   00733 126665R        JMP COMPL,I   GO TO ADDR SPECIFIED IN BFIN CALL
0548*
0549   00734 000022  SELBF  DEF BF
0550*
0551* BUOY DATA DRIVER
0552*
0553   00735 000000  BDLOC  NOP           ADDRESS
0554   00736 000000  BDIN   NOP
0555   00737 016001X        JSB .ENTR
0556   00740 000735R        DEF BDLOC
0557   00741 103100         CLF 0         INTERRUPTS OFF - SHOULD TAKE 6 MS
0558   00742 063301R        LDA =B120000  SET UP DMA
0559   00743 032770R        IOR SELBD
0560   00744 102606         OTA 6
0561   00745 106702         CLC 2
0562   00746 062735R        LDA BDLOC
0563   00747 033247R        IOR =B100000
0564   00750 102602         OTA 2
0565   00751 102702         STC 2
0566   00752 063302R        LDA =D-72
0567   00753 102602         OTA 2
0568   00754 103723         STC BD,C      ENABLE BUOY DATA INTERFACE
0569   00755 103706         STC 6,C       START DMA
0570   00756 002404         CLA,INA
0571   00757 102623         OTA BD        START EXTERNAL DEVICE
0572   00760 102306         SFS 6         WAIT FOR COMPLETION
0573   00761 026760R        JMP *-1
0574   00762 002400         CLA           STOP INTERFACE
```

```
0575  00763 102623         OTA BD
0576  00764 106723         CLC BD
0577  00765 106706         CLC 6           STOP DMA
0578  00766 102100         STF 0           INTERRUPTS ON
0579  00767 126736R        JMP BDIN,I      RETURN
0580  00770 000023  SELBD  DEF BD
0581*
0582* CLOCK DRIVER
0583*
0584  00771 000000  TADD   NOP             ADDRESS FOR TIME (2 WORDS)
0585  00772 000000  TDIN   NOP
0586  00773 016001X        JSB .ENTR
0587  00774 000771R        DEF TADD
0588  00775 063271R        LDA =D-3        SET CNTR FOR MAX 4 READS
0589  00776 073110R        STA CNTR
0590  00777 017017R        JSB TIN         INPUT BOTH WORDS TO TEMP
0591  01000 104400         DST TEMP
      01001 001106R
0592  01002 017017R TDIN1  JSB TIN         INPUT BOTH WORDS TO CALLER'S LOC
0593  01003 104400         DST TADD,I
      01004 100771R
0594  01005 053106R        CPA TEMP        WORD 1 SAME AS PREVIOUS?
0595  01006 002001         RSS
0596  01007 027012R        JMP TDIN2       NO
0597  01010 057107R        CPB TEMP+1      WORD 2 SAME AS PREVIOUS?
0598  01011 126772R        JMP TDIN,I      YES, RETURN
0599  01012 104400  TDIN2  DST TEMP        NO, STORE NEW IN OLD
      01013 001106R
```

```
01014 037110R        ISZ CNTR      INCREMENT COUNTER
01015 027002R        JMP TDIN1     LT 4 SAMPLES, TRY AGAIN
01016 027065R        JMP ERR13     ERROR, NO 2 SAMPLES ALIKE IN 4
01017 000000  TIN    NOP           INPUT SUBROUTINE
01020 006404         CLB,INB
01021 106624         OTB TIME      GATE HOURS
01022 102524         LIA TIME
01023 067242R        LDB =D2
01024 106624         OTB TIME      GATE MIN, SEC
01025 106524         LIB TIME
01026 003000         CMA
01027 007000         CMB
01030 127017R        JMP TIN,I     RETURN TO TDIN
*
* S INTERRUPT DRIVER (WAIT FOR INTERRUPT AND RETURN)
*
01031 000000  SWAIT  NOP
01032 103100         CLF 0         INTERRUPTS OFF
01033 103724         STC TIME,C
01034 102324         SFS TIME      WAIT FOR INTERRUPT
01035 027034R        JMP *-1
01036 106724         CLC TIME      TURN OFF INTERFACE
01037 102100         STF 0         INTERRUPTS ON
01040 127031R        JMP SWAIT,I   RETURN
*
01041 102001  ERR01  HLT 1
01042 027041R        JMP *-1
01043 102002  ERR02  HLT 2
01044 027043R        JMP *-1
01045 102003  ERR03  HLT 3
01046 027045R        JMP *-1
01047 102004  ERR04  HLT 4
```

```
0632  01050 027047R        JMP *-1
0633  01051 102005   ERR05 HLT 5
0634  01052 027051R        JMP *-1
0635  01053 102006   ERR06 HLT 6
0636  01054 027053R        JMP *-1
0637  01055 102007   ERR07 HLT 7
0638  01056 027055R        JMP *-1
0639  01057 102010   ERR10 HLT 10B
0640  01060 027057R        JMP *-1
0641  01061 102011   ERR11 HLT 11B
0642  01062 027061R        JMP *-1
0643  01063 102012   ERR12 HLT 12B
0644  01064 027063R        JMP *-1
0645  01065 102013   ERR13 HLT 13B
0646  01066 027065R        JMP *-1
0647  01067 102014   ERR14 HLT 14B
0648  01070 027067R        JMP *-1
0649  01071 102015   ERR15 HLT 15B
0650  01072 027071R        JMP *-1
0651*
0652  01073 000000   N     NOP
0653  01074 000000   L     NOP
0654  01075 000000   K     NOP
0655  01076 000000   M     NOP
```



```
0656  01077 000000   D     NOP
```

```
0657  01100 000000  S      NOP
0658  01101 000000  #BMS   NOP
0659  01102 000000  TOTPG  NOP
0660  01103 000000  PING   NOP
0661  01104 000000  BLOCK  NOP
0662  01105 000000  BCTR   NOP
0663  01106 000000  TEMP   BSS 2
0664  01110 000000  CNTR   NOP
0665  01111 000000  ANGL1  NOP
0666  01112 000000  FSTB#  NOP
0667  01113 000000  FMTB   NOP
0668  01114 000000  FBCTR  NOP
0669  01115 000000  NFTSW  NOP
0670  01116 000000  PTR1   NOP
0671  01117 000000  PTR2   NOP
0672  01120 000000  AA     NOP
0673  01121 000051  XFER   OCT 51
0674  01122 102030  HLT30  HLT 30B
0675  01123 000023  CYLAN  OCT 23            CYL NO. OF ANALYSIS SWAP AREA
0676  01124 000031  CYLCO  OCT 31            CYL NO. OF COLLECTION SWAP AREA
0677  01125 001240R OBUFA  DEF OBUF
0678  01126 021240R BUF1A  DEF BUF1
0679  01127 041240R BUF2A  DEF BUF2
0680  01130 000000  DTA72  BSS 72
0681  01240 000000  OBUF   BSS 8192
0682  21240 000000  BUF1   BSS 8192
0683  41240 000000  BUF2   BSS 8192
0684  00017         DISCD  EQU 17B
0685  00020         DISCC  EQU 20B
0686  00013         TBG    EQU 13B
0687  00022         BF     EQU 22B
0688  00023         BD     EQU 23B           (AVIONICS)
0689  00024         TIME   EQU 24B
0690  00000         A      EQU 0
0691  00001         B      EQU 1
```

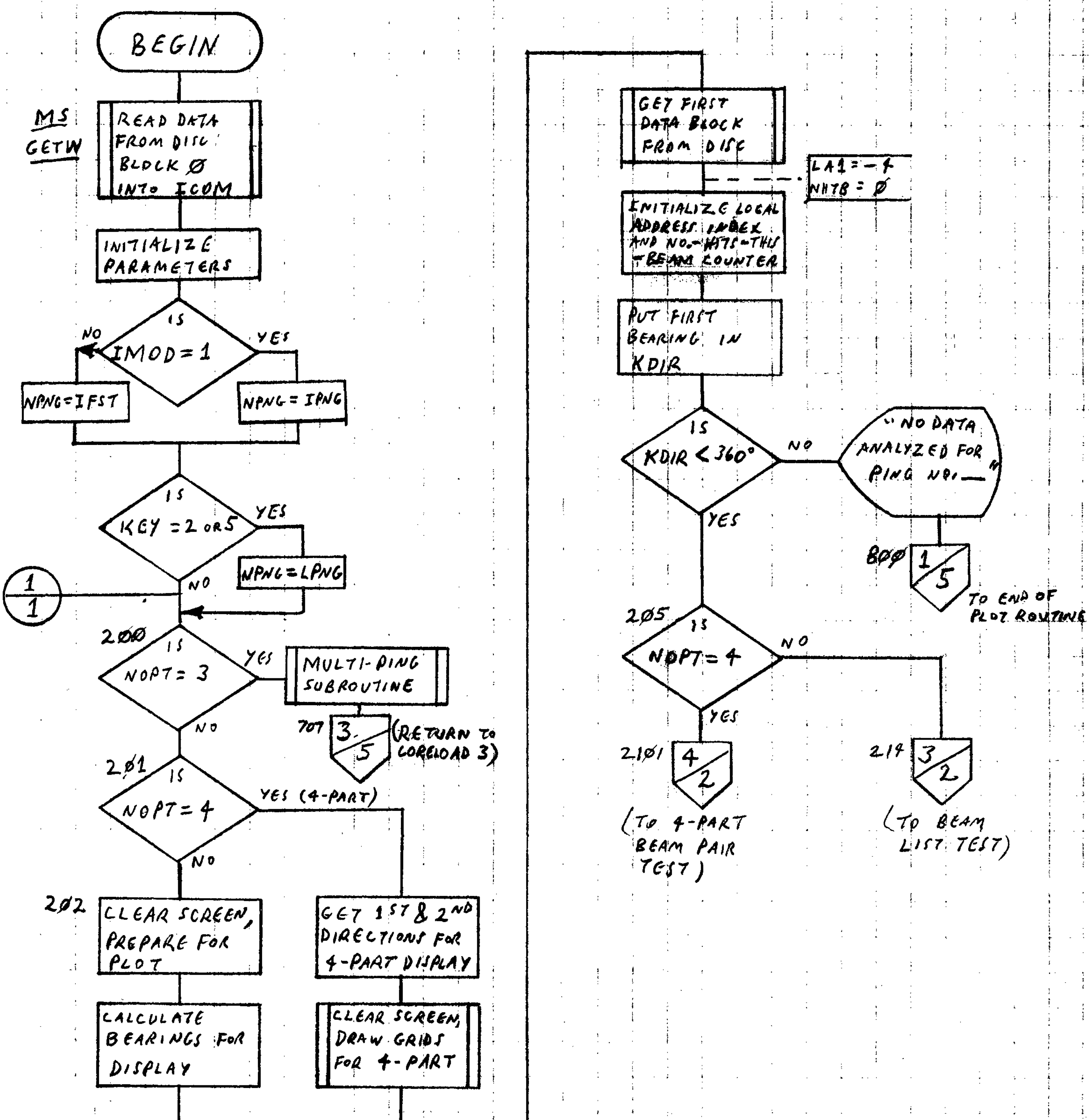
```
61240 051525
61241 052002
61242 000002
61243 177777
61244 000012
61245 177400
61246 177770
61247 100000
61250 055000
61251 000002
61252 040000
61253 000000
61254 040000
61255 000004
61256 000550
61257 055000
61260 000012
61261 040000
61262 000002
61263 000020
```

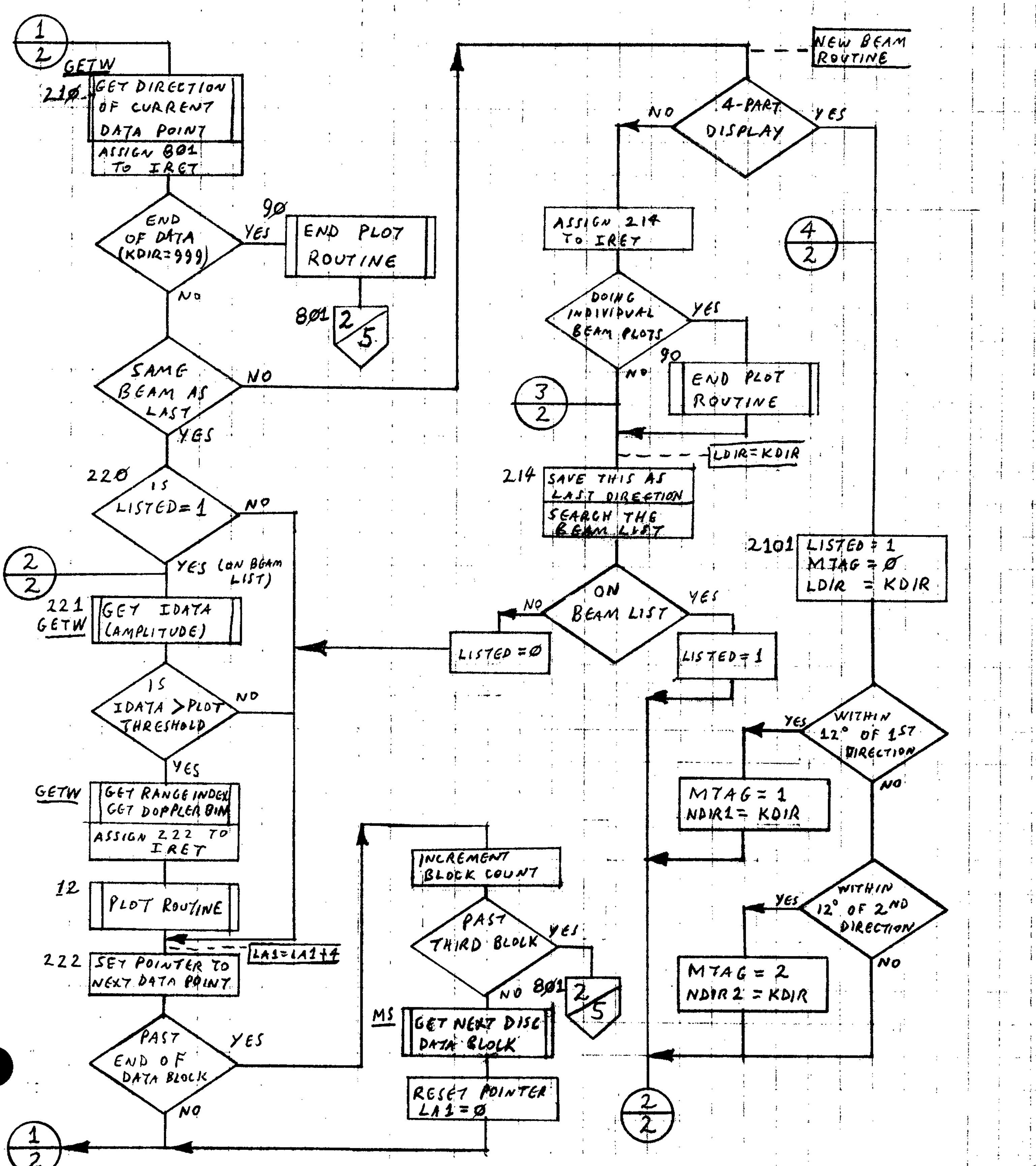


```
61264 177757
61265 177760
61266 000021
61267 000055
61270 177230
```

```
          61271 177775
          61272 177000
          61273 001000
          61274 000003
          61275 000200
          61276 014000
          61277 000060
          61300 177750
          61301 120000
          61302 177670
0692                           END Y0001
**  NO ERRORS *TOTAL **RTE ASMB 750420**
```

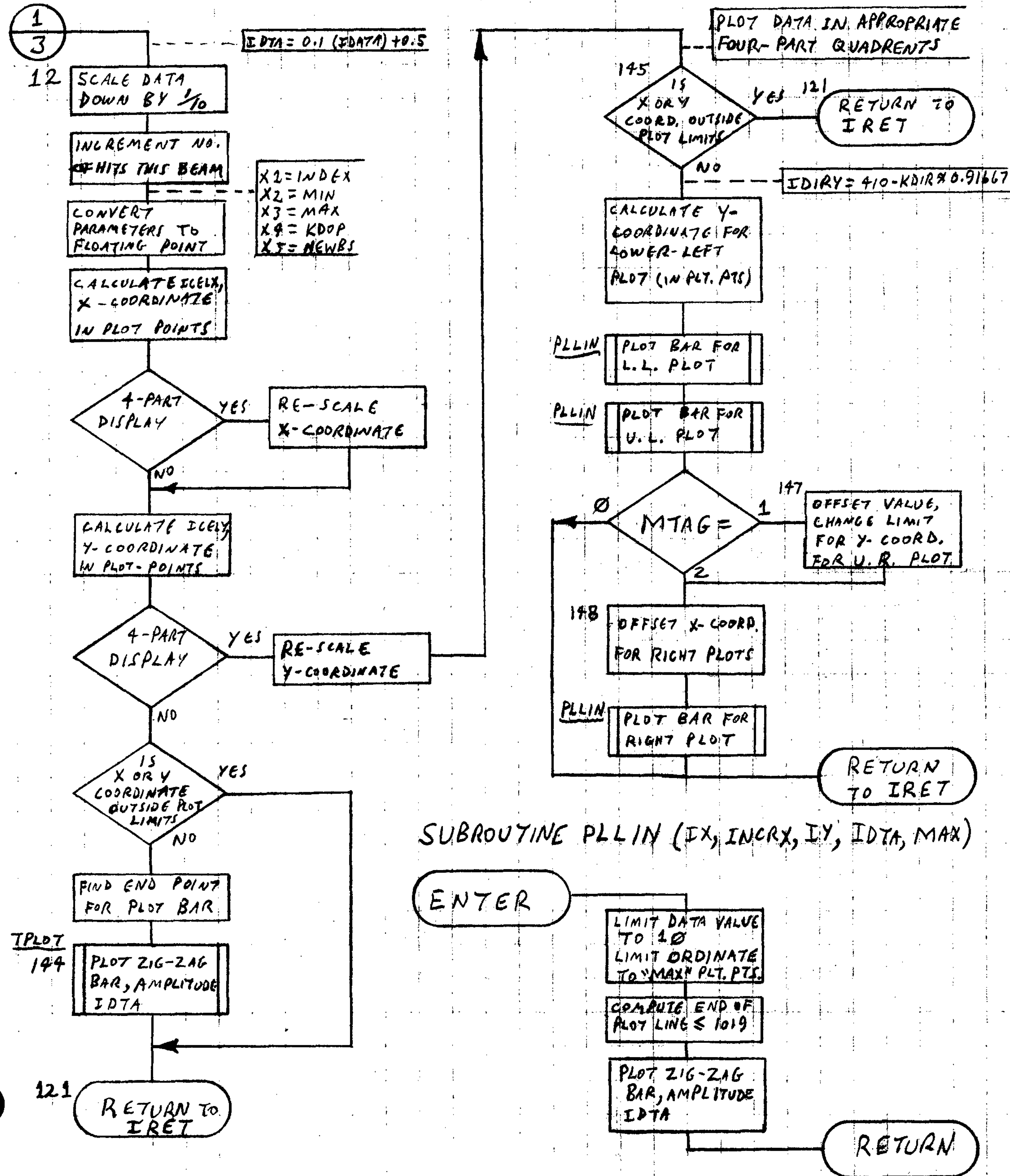
# Y1ØØ7 PLOT ROUTINE



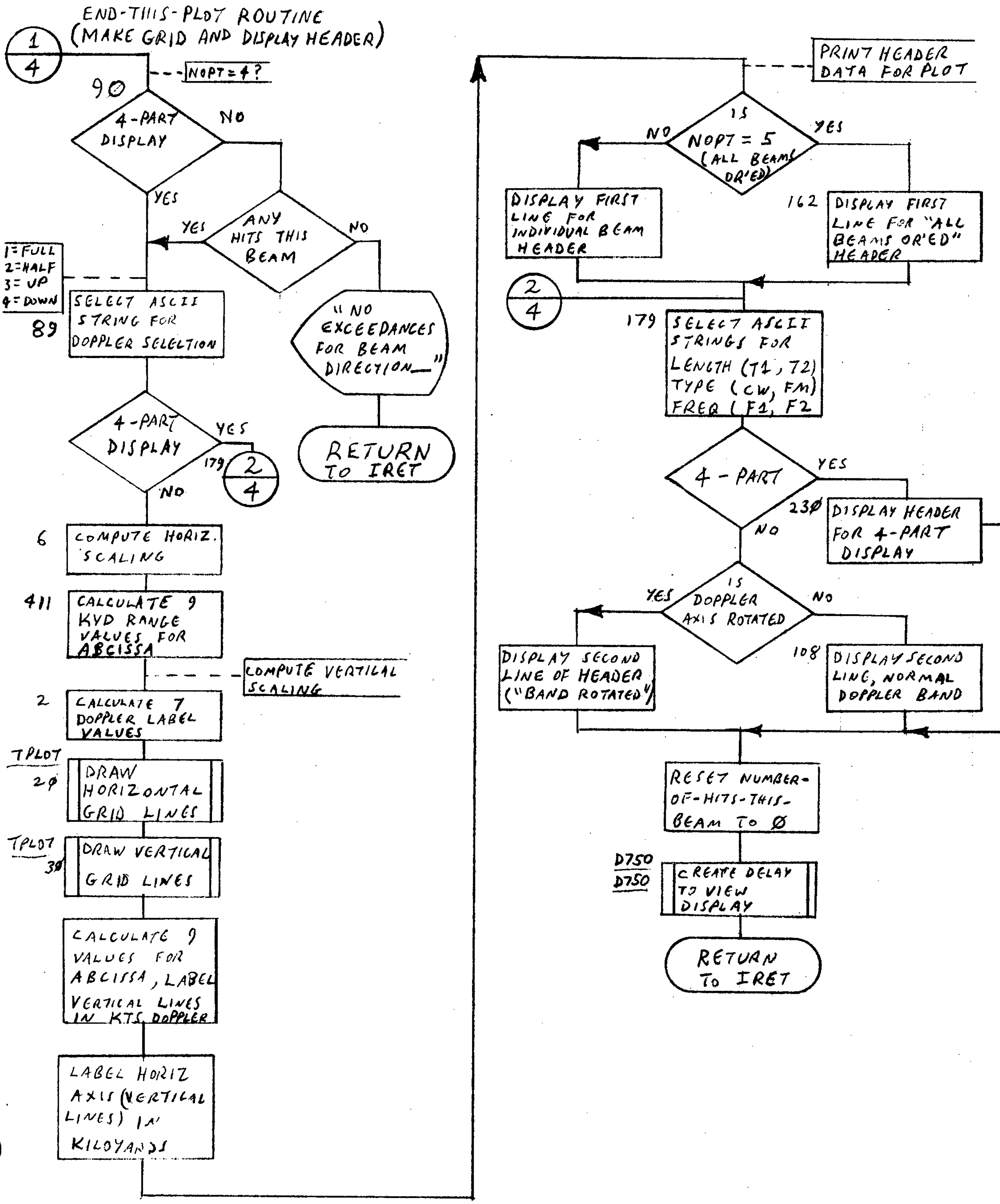



Y1007 PLOT ROUTINE (SHEET 2 OF 5)

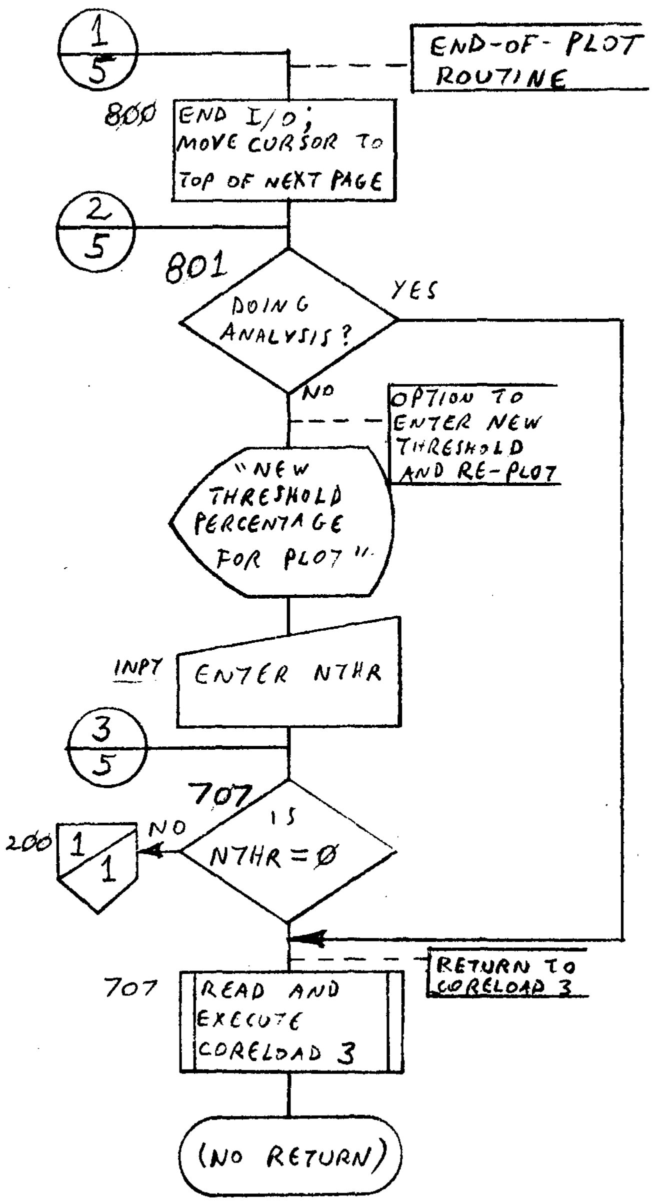
ROUTING TO PLOT A DATA POINT



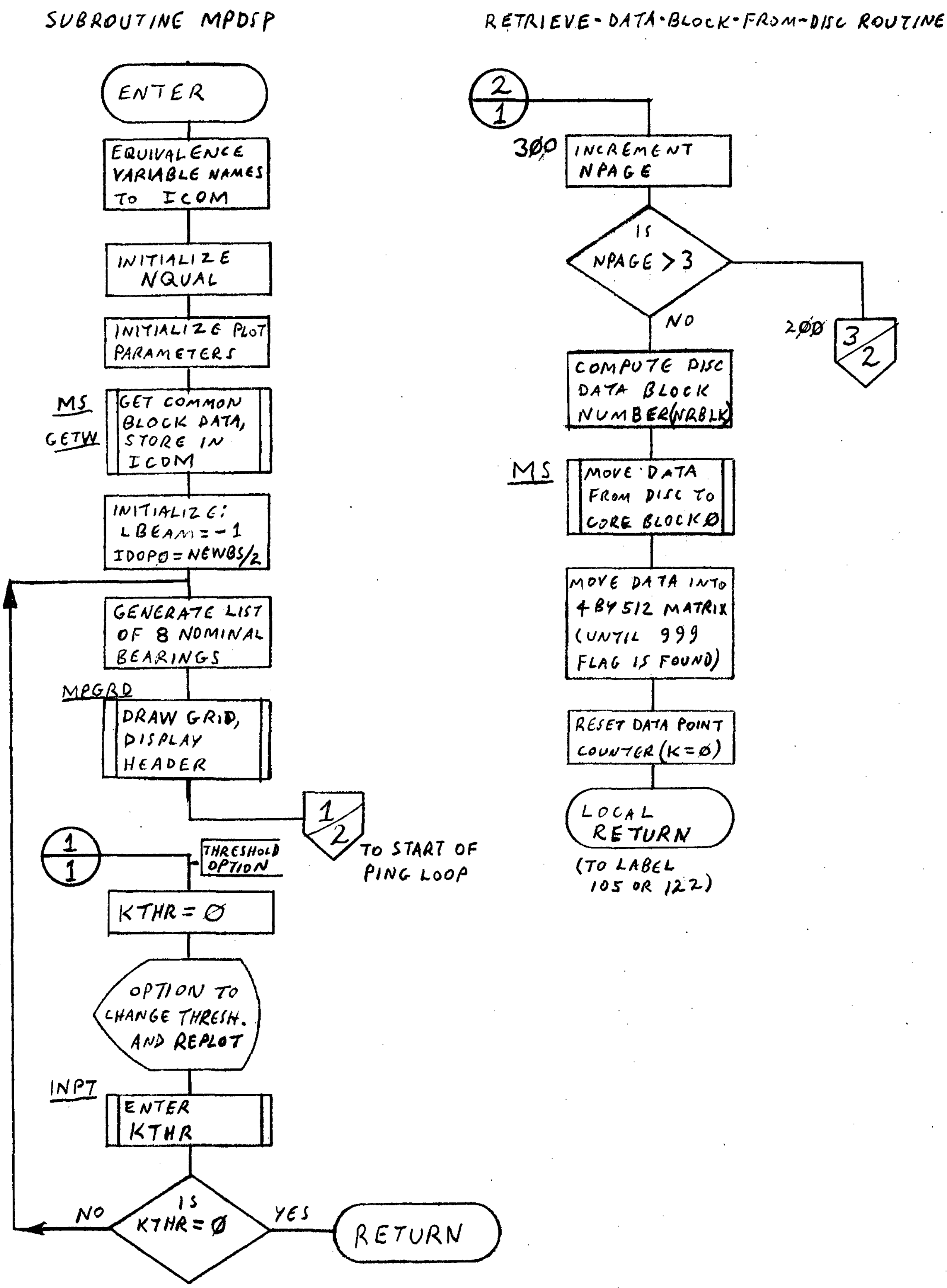

Y1007 PLOT ROUTINE (SHEET 3 OF 5)



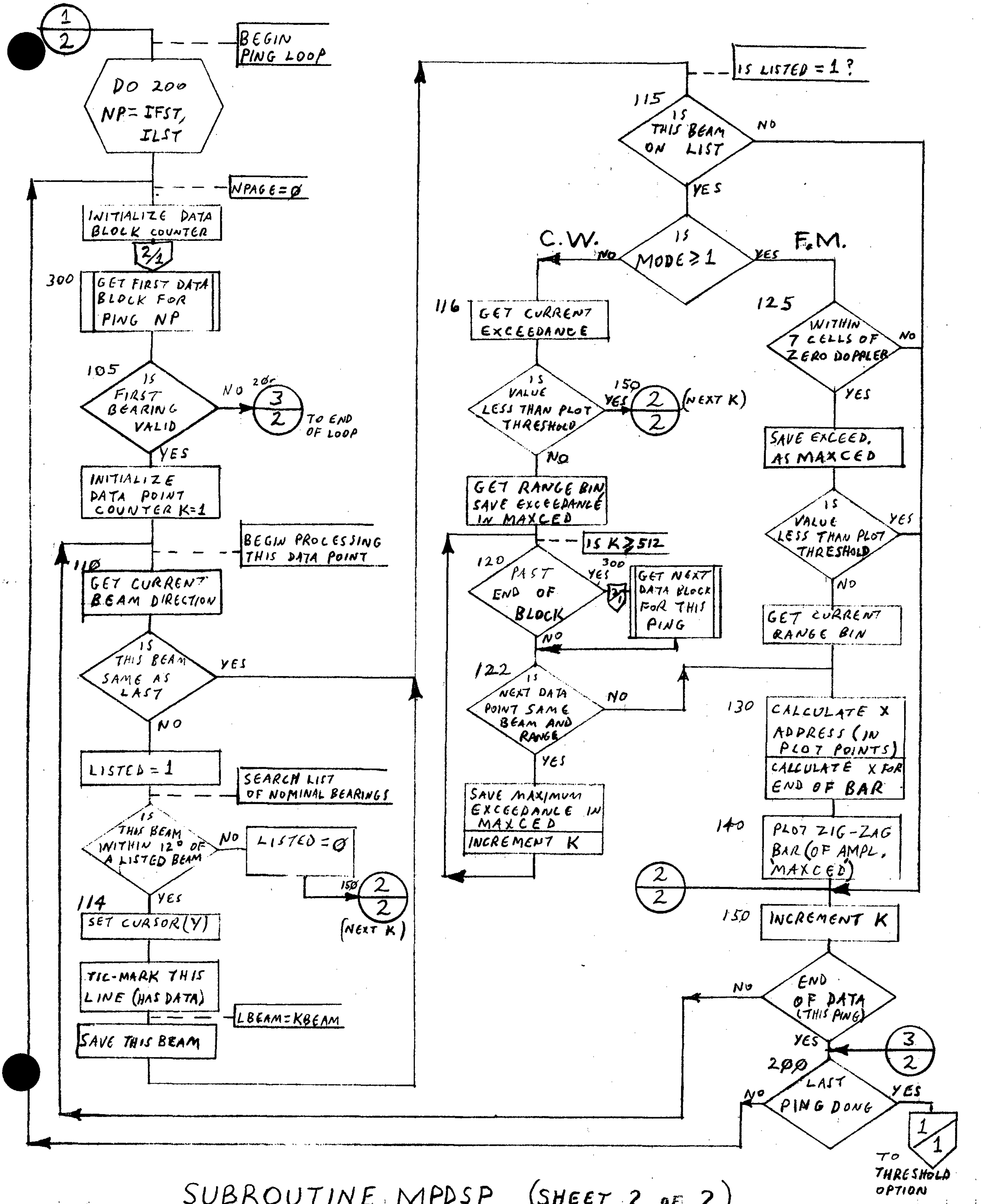

Y1007 PLOT ROUTINE (SHEET 4 OF 5)




Y1ØØ7 PLOT ROUTINE (SHEET 5 OF 5)

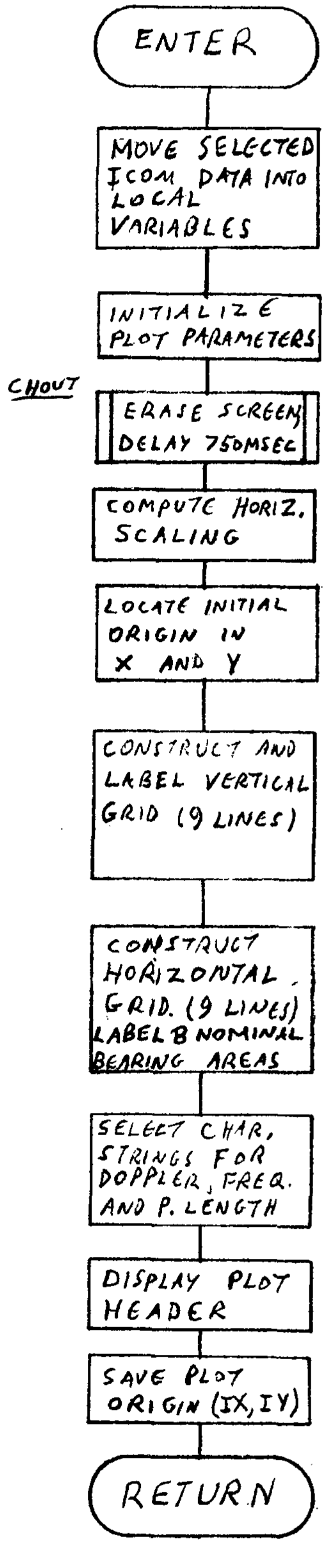



SUBROUTINE MPDSP (SHEET 1 OF 2)

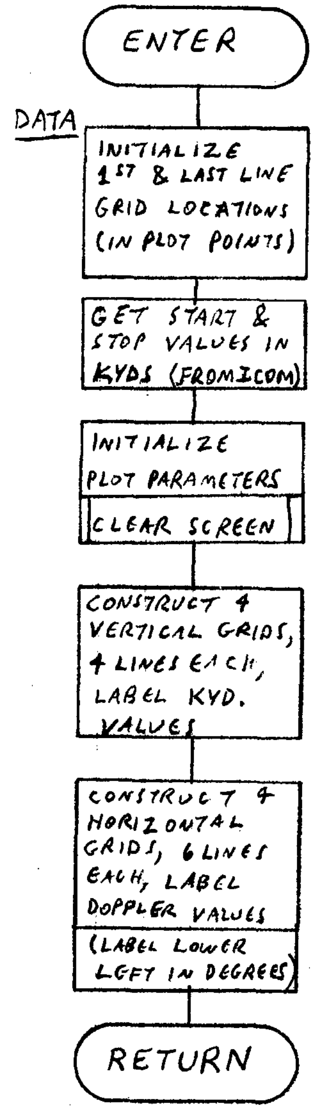



SUBROUTINE MPDSP (SHEET 2 OF 2)

SUBROUTINE MPGRD (ICOM, LISTBM)




SUBROUTINE FPGRD (ICOM)




GRID CONSTRUCTION SUBROUTINES

(SHEET 1 OF 1)

MP007A T=00004 IS ON CR00001 USING 00138 BLKS R=0796

```
FTN4
      SUBROUTINE Y1007
C                                    LATEST MODIFICATION:  2/10/78 (WPB)
C    ***************************************************************
C    *                                                             *
C    *  TITLE: OLAAS PLOT DISPLAYS SUBROUTINES                     *
C    *                                                             *
C    *  MODULE: Y1007        SOURCE: MP1007       VERSION:  X02    *
C    *                       OBJECT: Y10070                        *
C    *                                                             *
C    *  AUTHOR: R.A. FULLERTON             1976                    *
C    *                                                             *
C    *  MODIFICATIONS:                                             *
C    *    1:  ADDING MULTI-PING DISPLAY                            *
C    *        11/30/77    C.R. MOOSE                               *
C    *    2:  CHANGE TO CALCULATE THE ANALYSIS THRESHOLD           *
C    *         1/21/78    W. BOWEN                                 *
C    *                                                             *
C    *  MODULE FUNCTION:                                           *
C    *       THIS SUBROUTINE IS USED TO PLOT ERAPS DETECTION       *
C    *  INFORMATION ON A TEKTRONIX 4012 DISPLAY.    PLOT           *
C    *  PARAMETERS ARE OBTAINED FROM THE ANALYSIS PART OF THE      *
C    *  PROGRAM (CORELOAD 1/3).                                    *
C    *                                                             *
C    *  CALLING SEQUENCE:     CALLED BY - Y1002 (ANALYSIS)         *
C    *                                                             *
C    *     CALL CLOAD (53B,51)                                     *
C    *                                                             *
C    ***************************************************************
```

```
C     ******************************************************************
C     *  COMMON AREA ASSIGNMENTS  (ICOM AND/OR DISC DATA BLOCK 0)     *
C     ******************************************************************
C     *                                                                *
C     *  (0)    JDIV: DIVISOR OF ORIGINAL BLOCK FOR DOPPLER EDITING    *
C     *  (1)    IBND: SPECIFIC DOPPLER ANALYSIS BAND (1-JDIV BANDS)    *
C     *  (2)   NEWBS: BLOCK SIZE OF DOPPLER BAND                       *
C     *  (3)     MIN: MINIMUM INDEX USED IN ANALYSIS BY RDATA PGM.     *
C     *  (4)     MAX: MAXIMUM INDEX USED IN ANALYSIS BY RDATA PGM.     *
C     *  (5)   IFRAC: ANALYSIS THRESHOLD PERCENTAGE SET BY USER        *
C     *  (6)    INCR: TIME INCREMENT IN MULTIPLES OF 0.192 SECONDS     *
C     *  (7)    KODE: HANNING EQUALS 1, NO HANNING EQUALS 0            *
C     *  (8)   KTYPE: RANGE, KILOYARDS EQUALS 1, TIME EQUALS 0         *
C     *  (9)    IPNG: PING NUMBER                                      *
C     * (10)    KNUM: NUMBER OF BEAMS USED IN ANALYSIS                 *
C     * (11)   KOFFS: TIME OFFSET VALUE                                *
C     * (12)    MODE: 0 = CW PULSE,  1 = FM UP,  2 = FM DOWN           *
C     * (13)   LNGTH: EITHER 2 OR 5 SECONDS                            *
C     * (14)     KT1: TIME, HOURS                                      *
C     * (15)     KT2: TIME, MINUTES                                    *
C     * (16)     KT3: TIME, SECONDS                                    *
C     * (17)    LFRQ: BUOY FREQUENCY, 1 = F1, 2 = F2                   *
C     * (18)    JDIV: DIVISOR OF ORIG. BLOCK (SAME AS WORD 0)          *
C     * (19)   NTATE: DOPPLER AXIS ROTATION, 1 = YES, 0 = NO           *
C     * (20)   NSHFT: FOR NTATE=1, NO. OF KNOTS TO SHIFT DOPPLER       *
C     * (21)            SPARE                                          *
C     * (22)     KEY: 0 = ON-LINE PLOT                                 *
C     *               1 = ANALYSIS OF PREVIOUSLY RECORDED DATA         *
C     *               2 = INDIVIDUAL BEAM DISPLAY                      *
C     *               3 = MULTI-PING DISPLAY                           *
C     *               4 = FOUR-PART DISPLAY                            *
C     * (23-32) ITH1: NUMBER OF WORDS STORED FOR EACH PING             *
C     * (33)                SPARE                                      *
C     * (34) MXPNGS: MAXIMUM NUMBER OF PINGS TO BE COLLECTED           *
C     * (35)    IFST: FIRST PING NO. FOR ANALYSIS                      *
```

```
C    * (36)    ILST: LAST PING NO. FOR ANALYSIS                          *
C    * (37)   IDPLR: DOPPLER BAND (1=FULL, 2=HALF, 3=UP, 4=DOWN)         *
C    * (38)   JSTRT: START RANGE (IN KYDS.)                              *
C    * (39)    JEND: END   RANGE (IN KYDS.)                              *
C    * (40)    JDIR: DIRECTION OF BEARING OF CENTER BEAM OR              *
C    *               BEAN NUMBER TO DISPLAY FOR FOUR PART DISPLAY        *
C    * (41)    IMOD: MODE (ON-LINE / OFF-LINE ANALYSIS)                  *
C    * (42-46)            SPARES                                         *
C    * (47)   INCRX: LENGTH OF ONE RANGE INCREMENT (IN PLOT POINTS)      *
C    * (48)    KTHR: CURRENT PERCENTAGE FOR PLOT THRESHOLD (1-100)       *
C    * (49-50)            SPARE                                          *
C    * (51-66) IHDR: HEADER FROM INPUT DATA TAPE                         *
C    * (67-69)             SPARES                                        *
C    * (70) NFTTAG: DUPLICATE OF NOT-FIRST-TIME SWITCH                   *
C    * (71-80)  IAVGEX: NUMBER OF EXCEDENCES FOR ALL 10 PINGS            *
C    * (81-90)   MIDBM: CENTER BEAM ANALYZED FOR ALL 10 PINGS            *
C    * (91-100)  NRBMS: NUMBER OF BEAMS ANALYZED FOR ALL 10 PINGS        *
C    * (101-109) IDATE: ALPHANUMERIC FIELD FOR DATE (18 CHAR.)           *
C    ****************************************************************

C    ****************************************************************
C    *  DEFINE ARRAYS, CONSTANTS, VARIABLES, & DATA                      *
C    ****************************************************************

        DIMENSION NQUAL(5),HORZ(9),VERT(7),NBM(16),IBCD(6),LABEL(2),
       +MASK(4),ICOM(127),IHDR(16),ITH(10),ITHNDX(10),LISTBM(16),
       +IAVGEX(10),MIDBM(10), NRBMS(10), IDATE(9), OFFSET(4,2),
       +PSCALE(2)

        EQUIVALENCE
       +(ICOM(  1),  IBND),(ICOM(  2), NEWBS),(ICOM(  3),   MIN),
       +(ICOM(  4),   MAX),(ICOM(  5), IFRAC),(ICOM(  6),  INCR),
       +(ICOM(  7),  KODE),(ICOM(  8), KTYPE),(ICOM(  9),  IPNG),
       +(ICOM( 10),  KNUM),(ICOM( 11), KOFFS),(ICOM( 12),  MODE),
       +(ICOM( 13), LNGTH),(ICOM( 14),   KT1),(ICOM( 15),   KT2),
```

```
     +(ICOM( 16),    KT3),(ICOM( 17),  LFRQ),(ICOM( 18),  JDIV),
     +(ICOM( 19), NTATE),(ICOM( 20), NSHFT),
     +(ICOM( 22),    KEY),(ICOM( 23),   ITH),
     +(ICOM( 34),MXPNGS),(ICOM( 35),  IFST),(ICOM( 36),  ILST),
     +(ICOM( 37), IDPLR),(ICOM( 38), JSTRT),(ICOM( 39),  JEND),
     +(ICOM( 40),  JDIR),(ICOM( 41),  IMOD),
     +                   (ICOM( 47), INCRX),(ICOM( 48),  KTHR),
     +                                      (ICOM( 51),  IHDR),
     +(ICOM( 70),NFTTAG),(ICOM( 71),IAVGEX),(ICOM( 81), MIDBM),
     +(ICOM( 91), NRBMS),(ICOM(101), IDATE)

      DATA ITHNDX/1,2,3,4,5,6,7,8,9,10/
      DATA MASK/17B,360B,7400B,170000B/
      DATA NQUAL/2048,100B,16383,45,0/,LISTBM/16*0/,NOPT/2/
      DATA  IERASE,IESCAP,IRESET,IDASH/14B,33B,37B,55B/
      DATA  OFFSET/104.,274.,444.,104.,719.,577.,719.,435./
      DATA  NDIR1, NDIR2, NDIR3, NDIR4/4*999/
      DATA  PSCALE/2.6592,2.216/,NDIR2,NDIR3/2*999/

C   ****************************************************************
C   *  OLAAS PLOT DISPLAY SUBROUTINE INITIAL ENTRY POINT           *
C   ****************************************************************

C   >>>>>>> DIAGNOSTIC >>  SENSE SWITCH 5 (ON) DUMP DATA BLOCK
      IF(ISSW(5) .LT. 0) CALL BRDB

C - - READ DATA FROM DISC BLOCK 0 TO CORE BLOCK 0
      CALL KYBD(2HMS, 31, 0)
      CALL KYBD(2HMS, 11, 0)

C - - - - - - - - - TRANSFER DATA INTO COMMON VECTOR (ICOM)
      DO 106  J = 1, 127
       CALL GETW(0, J, ICOM(J), II)
  106 CONTINUE
```

```
C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 10 (ON) WRITES ICOM
        IF(ISSW(10) .LT. 0) WRITE(1, 9106) ICOM
 9106   FORMAT("07", 10I7)
        IF(ISSW(10) .LT. 0) WRITE (1, 9998)
 9998   FORMAT(" ")
        IF(ISSW(10) .LT. 0) PAUSE 15

C - - - - - - - - INITIALIZE NEW PARAMETERS FOR THIS PLOT
       NSEP   = 112
       IX     = 1023 - 8 * NSEP
       IY     =  780 - 6 * NSEP

           IF (ISSW(14) .LT. 0) GO TO 107
       THRES = ICOM( 5) * 1E-6
   107 NTHR   = KTHR
       NDIR   = JDIR
       NBEAMS= KNUM
       NOPT   = KEY
       NPNG   = IPNG
  200  IF(NOPT .NE. 3)                GO TO 201

C - - - - - - - -  TRANSFER TO MULTI-PING ROUTINE
       CALL MPDSP
                                      GO TO 707

  201  IF(NOPT .NE. 4)                GO TO 202

C - - - - - - - - FOR 4-PART, GET FIRST AND SECOND NOMINAL DIRECTION
       NDIR1 = NDIR - 12
       IF(NDIR1 .LT. 0) NDIR1 = NDIR1 + 360
       NDIR2  = MOD(NDIR + 12, 360)

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 9 (ON)
       IF(ISSW(9) .LT. 0) WRITE(1, 9108)NDIR1,NDIR2,NDIR
 9108   FORMAT("NDIR1&2 =",3I6)
```

```
      IF(ISSW(9) .LT. 0) PAUSE 24

C - - - - - - - -  DRAW GRID FOR FOUR-PART DISPLAY
      CALL FPGRD(ICOM)
                                    GO TO 2035

C - - - - - - - - - - -  CLEAR SCREEN, PREPARE FOR PLOT

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 9 (ON)
 202  IF(ISSW(9) .LT. 0) PAUSE 1

      CALL TPLOT(0,0,0)
      CALL CHOUT(IESCAP)
      CALL CHOUT(IERASE)
      CALL D750

C - - - - - - - - - CALCULATE BEARINGS OF DESIRED BEAMS FOR DISPLAY
      BRG = NDIR - 22.5*IFIX((NBEAMS-1)/2.) -11.25*MOD(NBEAMS-1,2)
      DO 203  N = 1, NBEAMS
       IF(BRG .GE. 360.) BRG = BRG - 360.
       IF(BRG .LT.   0.) BRG = BRG + 360.
       LISTBM(N) = IFIX(BRG + 0.5)
 203  BRG = BRG + 22.5

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 10 (ON)
      IF(ISSW(10) .LT. 0) WRITE(1, 9201) (LISTBM(I), I = 1, NBEAMS
9201                    FORMAT("BEAMS:",8I6)

C - - - - - - GET FIRST DATA BLOCK FROM DISC,
C             SET LOCAL ADDRESS AND NUMBER OF HITS THIS BEAM TO 0.
 2035 KTBLK = 1
      NRBLK = 3*(NPNG-1) + 1
      CALL PUTQ(0, NQUAL)
      CALL KYBD(2HMS, 31, NRBLK)
      CALL KYBD(2HMS, 11, 0)
```

```
      LA   = 0
      NHTB = 0

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 9 (ON)
      IF(ISSW(9) .LT. 0) WRITE(1, 9990)KTBLK,NRBLK,LA,LISTBM(1)

C - - - - - - - - - - - GET FIRST DIRECTION, GO TO BEAM LIST TEST
 204  CALL GETW(0, 0, KDIR, II)
      IF(KDIR .LE. 360)                GO TO 205
      WRITE(1, 9214)  NPNG, KDIR
9214  FORMAT("NO DATA ANALYZED FOR PING NO.", I3, ". BEARING =", I
                                       GO TO 800
 205  IF(NOPT .EQ. 4)                  GO TO 2101
                                       GO TO 214

C - - - - - - - GET DIRECTION OF CURRENT DATA POINT.  END OF DATA?
 210  CALL GETW(0, LA, KDIR, II)

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 6 (ON)
      IF(ISSW(6) .LT. 0) WRITE(1, 9210)KDIR, LDIR, LA
9210   FORMAT(50X, "KDR", 3I5)

      ASSIGN 801 TO IRET
      IF(KDIR .EQ. 999)                GO TO 90

C - - - - - - - - IF THIS BEAM IS SAME AS LAST, GO TEST AMPLITUDE
      IF(KDIR .EQ. LDIR)               GO TO 220

C NEW BEAM - - - - - - - - - - NEW BEAM  - - - - - - - - NEW BEAM - - -

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 9 (ON)
       IF(ISSW(9) .LT. 0) WRITE(1, 9211)LDIR, KDIR
9211   FORMAT("LAST/NEW DIR.",2I5)
```

```
C - - - - - - - - - -  IF FOUR-PART, IS THIS ONE OF TWO DESIRED BEAMS?
          IF(NOPT .NE. 4)                 GO TO 212
 2101   LISTED = 1
        LDIR = KDIR
        MTAG = 0
          IDIF = IABS(KDIR - NDIR1)
          IF(IDIF .GT. 12 .AND. IDIF .LT. 348)  GO TO 211
        MTAG = 1
        NDIR3 = KDIR
                                                GO TO 211
  211     IDIF = IABS(KDIR - NDIR2)
          IF(IDIF .GT. 11 .AND. IDIF .LT. 349)  GO TO 221
        MTAG = 2
        NDIR4= KDIR
                                                GO TO 221

C - - - - -IF PLOTTING INDIVIDUAL BEAMS, END PLOT FOR PREVIOUS BEAM
  212   ASSIGN 214 TO IRET
        IF(NOPT .LE. 2)           GO TO 90
C                      > > > > > > > > > > > > >

C - - - - - SAVE THIS DIRECTION.     IS THIS BEAM ON THE LIST?
  214   LDIR  = KDIR
        LISTED= 1
        DO 216  KTB = 1, NBEAMS
         IDIF = IABS(LISTBM(KTB) - LDIR)
         IF(IDIF .LE. 12 .OR. IDIF .GE. 348)   GO TO 221
  216   CONTINUE
C                NOT IN BEAM LIST
        LISTED = 0
                                  GO TO 222

C - - - - -  FOUND IN BEAM LIST?  GET DATA VALUE, TEST THRESHOLD
  220   IF(LISTED .EQ. 0)        GO TO 222
  221   CALL GETW(0, LA+2, IDATA, II)
```

```
      IF(IDATA .LE. NTHR)     GO TO 222

C - - - - - - - - - - - - - OVER THRESHOLD  -  GO TO PLOT ROUTINE
      CALL GETW(0, LA+1, INDEX, II)
      CALL GETW(0, LA+3,  KDOP, II)

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 8 (ON)
      IF(ISSW(8) .LT. 0)WRITE(1,9990)KDIR,INDEX,IDATA,
     *                          KDOP,LA,NTHR,MTAG

      ASSIGN 222 TO IRET
                                    GO TO 12
C                                  >>>>>>>> > > > > > > > >

C - - - - - - - INCREMENT LOCAL ADDRESS, TEST FOR END OF BLOCK
 222  LA = LA + 4

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 4 (ON)
       IF(ISSW(4) .LT. 0) WRITE(1, 9222)LA
9222    FORMAT(" LA = ",I5)

      IF(LA .LE. 2044)              GO TO 210

C - - - INCREMENT BLOCK COUNT, GET NEXT BLOCK FROM DISC, RESET LA
      KTBLK = KTBLK + 1
      IF(KTBLK .GT. 3)              GO TO 801
      NRBLK = 3*(NPNG-1) + KTBLK
      CALL KYBD(2HMS, 31, NRBLK)
      CALL KYBD(2HMS, 11, 0)
      LA = 0

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 5 (ON)
       IF(ISSW(5) .LT. 0) WRITE(1, 9990)NRBLK, KTBLK, LA

                                    GO TO 210
```

```
C %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

C - -COMPRESS DATA VALUE FOR PLOTTING, INCR. NO. OF HITS THIS BEAM
   12  IDTA = IFIX(0.1 * IDATA + 0.5)

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 1 (ON)
         IF(ISSW(1) .LT. 0) WRITE(2,989)KDIR,INDEX,IDTA,KDOP,MTAG
  989    FORMAT(1X, 5I9)

        NHTB = NHTB + 1

C > > > > CONVERT TO FLOATING POINT
        X1 = INDEX
        X2 = MIN
        X3 = MAX
        X4 = KDOP
        X5 = NEWBS
        ICELX = (INDEX - MIN)*896.0 / (MAX - MIN) + 127.5
        IF (NOPT .EQ. 4) ICELX = ICELX * 0.4911 + 1.6

C > > > > DOPPLER SCALING - COMPUTE Y COORDINATE
        ICELY = OFFSET(IDPLR,LFRQ) + PSCALE(LFRQ) * X4
        IF (NOPT .EQ. 4) ICELY = ICELY * 0.9821 + 179.5

C > > > > BRANCH TO PROPER LIMIT CHECK AND PLOT (1-BEAM OR 4-PART)
        IF (NOPT .EQ. 4) GO TO 145
        IF(ICELX .LT. IX .OR. ICELX .GT. 1019) GO TO 121
        IF(ICELY .LT. IY .OR. ICELY .GT. 780) GO TO 121

C  ******************************************************************
C  *  PLOT THE DETECTIONS                                           *
C  ******************************************************************

        IF (IDTA .GT. 10) IDTA = 10
        M = 1
```

```
      MY = ICELY
 2144 IF (MY .GT. 780) GO TO 2147
      CALL TPLOT (0,ICELX,MY)
      CALL CHOUT (37B)
      CALL CHOUT (55B)
      MY = MY + 2
      M = M + 1
      IF (M .LE. IDTA)  GO TO 2144
 2147 CONTINUE

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 9 (ON)
      IF(ISSW(9) .LT. 0)  WRITE(1, 9121)ICELX,MY,LA,IDTA
 9121 FORMAT("PLT",4I5)

C > > > > RETURN TO CALLER
  121   GO TO IRET

C > > > > PLOT DETECTIONS ON 4-PART DISPLAY
  145 IF (ICELX .LT. 60 .OR. ICELX .GT. 500) GO TO 121
      IF (ICELY .LT.450 .OR. ICELY .GT. 780) GO TO 121
      IF (IDTA .GT. 10) IDTA = 10
      IDIRY = 410 - KDIR * 0.91667
      CALL PLLIN (ICELX,INCRX,IDIRY,IDTA,420)
      CALL PLLIN (ICELX,INCRX,ICELY,IDTA,780)
      MAXHT=780
      IF (MTAG.EQ.1) GO TO 148
      IF (MTAG.EQ.2) GO TO 147
                     GO TO IRET
  147 MAXHT=410
      ICELY=ICELY-370
  148 ICELX=ICELX+510
      CALL PLLIN (ICELX,INCRX,ICELY,IDTA,MAXHT)
                     GO TO IRET
C       < < < < < < < < < < < < <
C XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

```
C > > > > END PLOT.  MAKE GRID AND PRINT HEADER.
   90  CALL CHOUT(IRESET)
       IF (NOPT .EQ. 4) GO TO 89
       IF(NHTB .LE. 0 .AND. LISTED .EQ. 1) WRITE(1, 9090) LDIR
9090    FORMAT(" NO EXCEEDENCES FOR BEAM DIRECTION", I4)
       IF(NHTB .LE. 0)          GO TO IRET

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 9 (ON)
       IF(ISSW(9) .LT. 0) WRITE(1, 9091)IX,IY,NSEP,KOFFS,KDIR,LA
9091    FORMAT("GRD",6I6)

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 8 (ON)
       IF(ISSW(8) .LT. 0)  PAUSE 400

   89  GO TO (91,92,93,94) IDPLR
91     LABEL(1) = 2HFU
       LABEL(2) = 2HLL
       GO TO 95
92     LABEL(1) = 2HHA
       LABEL(2) = 2HLF
       GO TO 95
93     LABEL(1) = 2HUP
       LABEL(2) = 2H
       GO TO 95
94     LABEL(1) = 2HDO
       LABEL(2) = 2HWN
95     CONTINUE

C > > > > COMPUTE HORIZONTAL SCALING
       IF (NOPT .EQ. 4) GO TO 179
       X7 = KOFFS
     6 DELTI = JEND - JSTRT
       DO 411 M= 1,9
   411 HORZ(M) = JSTRT + (M-1) * DELTI / 8
```

```
C > > > > COMPUTE VERTICAL SCALING
      DO 2 I = 1,7
      J = I - 1
    2 VERT(I) = 30.0 - (J * 10.0)

C > > > > CONSTRUCT GRID
      DO 20 J=1,7
      LY=IY + (J-1)*NSEP
      CALL TPLOT(0,IX,LY)
   20 CALL TPLOT(1,1023,LY)

      DO 30 M=1,9
      LX=IX + (M-1)*NSEP
      CALL TPLOT(0,LX,IY)
   30 CALL TPLOT(1,LX,780 )

C > > > > LABEL VERTICAL AXIS
      DO 19 L = 1,7
      J = L - 1
      LY = 772 - (J*NSEP)
      IF(L .EQ. 1)LY = 780
      IF(L .EQ. 7) LY = LY + 12
      CALL TPLOT(0,55,LY)
      CALL CHOUT(IRESET)
 19   WRITE(2,22)VERT(L)
 22   FORMAT(F5.1)

C > > > > LABEL HORIZONTAL AXIS
      CALL TPLOT(0,91,85)
      CALL CHOUT(IRESET)
      WRITE(2,406)(HORZ(I),I=1,9)
  406 FORMAT(7(F5.1, 3X),F5.1,1X,F5.1)
      CALL TPLOT(0,0,62)
      CALL CHOUT(IRESET)
```

```
C * * * * * * * * * * *  PRINT HEADER DATA FOR PLOT * * * * * * *
C      CHECK FOR SINGLE OR MULTIPLE BEAM CASE (FOR HEADER INFO)
       KPNG = NPNG
       IF(NOPT .EQ. 5) GO TO 162
    37 WRITE (2,151) NPNG,LDIR,LABEL,IFRAC
   151 FORMAT ("PING NO:",I3,2X,"BEARING:",I4,2X,"DOPPLER: ",
      +2A2,5X,"ANALYSIS THRESHOLD: ",I5,"%")
       GO TO 179

C > > > > MULTIPLE BEAM CASE
   162 WRITE (2,163) NPNG,LABEL
   163 FORMAT ("PING NO:",I3,2X,"DOPPLER: ",2A2,2X,"** ALL"
      +" BEAMS OR'ED **")
   179                     LEN  = 2HT2
                           LFRE = 2HF2
                           LTYP = 2HCW
       IF(MODE .GT. 0) LTYP = 2HFM
       IF(LFRQ .EQ. 1) LFRE = 2HF1
       IF(LNGTH .EQ. 2) LEN = 2HT1
       IF (NOPT .EQ. 4) GO TO 230
       IF(NTATE .EQ. 0) GO TO 108

       WRITE (2,181) LTYP,LEN,LFRE,KODE,NTHR,NSHIFT,KT1,KT2,
      +KT3,IDATE
   181 FORMAT ("SIGNAL:",2X,A2,2X,A2,2X,A2,2X,"HANNING:",I2,18X,
      +"PLOT THRESHOLD:",I3,/,"(BAND ROTATED -",I3," KNOTS)",13X,
      +"TIME: ",I2,":",I2,":",I2,3X,9A2)
       GO TO 233
   108 WRITE (2,112) LTYP,LEN,LFRE,KODE,NTHR,KT1,KT2,KT3,IDATE
   112 FORMAT ("SIGNAL:",2X,A2,2X,A2,2X,A2,2X,"HANNING:",I2,18X,
      +"PLOT THRESHOLD:",I3,/40X,"TIME: ",I2,":",I2,":",I2,3X,9A2)
       GO TO 233

C > > > > FOUR-PART DISPLAY CASE
   230 CALL TPLOT(0,5,35)
```

```
      CALL CHOUT(IRESET)
      WRITE(1,9230) NPNG,LABEL,NTHR,NDIR3,NDIR4,LTYP,LEN,
     +LFRE,KODE,KT1,KT2,KT3,IDATE
 9230 FORMAT ("PING NO:",I3,2X,"DOPPLER: ",2A2,2X,"PLOT"
     +" THRESHOLD:",I3,2X,"RIGHT PLOT BRGS:",I4,"/",I3,/
     + "SIGNAL:",2X,A2,2X,A2,2X,A2,2X,"HANNING:"I2,8X,
     + "TIME: ",I2,":",I2,":",I2,3X,9A2)

C > > > > CREATE DELAY FOR VIEWING DISPLAYS
  233 CALL D750
      CALL D750
      NHTB =  0

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 7 (ON)
       IF(ISSW(7) .GE. 0)   GO TO IRET


C * * * * * * * *  END OF PLOTS.  RETURN TO CORELOAD 3
 800  CALL ENDIO
      CALL TPLOT(0, 0, 0)
      CALL CHOUT(IRESET)
      WRITE(1, 9998)
 801  CONTINUE

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 2 (ON)
      IF(ISSW(2) .LT. 0)    PAUSE 4

C     IF(IMOD .EQ. 1 .OR. NOPT .EQ. 1) GO TO 707

C - - - - - OPTIONS FOR PLOT OR NEW THRESHOLD - - -
      NTHR = 0
      WRITE(1, 9802)
 9802 FORMAT(" NEW THRESHOLD PERCENTAGE FOR PLOT (1 - 100)",/
     *  " (1% DISPLAYS ALL ANALYZED DATA,  100% = NONE)"/
     *  "  TYPE 0 TO RETURN TO ANALYSIS PROGRAM")
```

```
      CALL INPT(NTHR, 0, 100)
      IF(NTHR .EQ. 0)          GO TO 707
                               GO TO 200

C - - - - - - - - - - - - - - - EXIT PROGRAM. GO TO CORELOAD 3
  707 CALL ENDIO
      CALL CLOAD(23B,51B)
  157 WRITE(2,158)
  158 FORMAT("***** PARAMETER ENTRY ERROR *****")
      CALL ENDIO
      CALL CLOAD(23B,51B)
9990  FORMAT(1X, 10I7)
C ##################  END OF THE PLOT ROUTINE  ###################
      END

      SUBROUTINE MPDSP
C                           JAN. 26, 1978
C  ***************************************************************
C  *  OLAAS MULTI-PING DISPLAY ROUTINE                           *
C  ***************************************************************

      DIMENSION ICOM(127),LISTBM(8),NQUAL(5),MAT(4, 512),IDATE(9)

      EQUIVALENCE (ICOM(2),NEWBS),(ICOM(3),MIN),(ICOM(4),MAX),
     +(ICOM( 6), INCR),(ICOM(11),KOFFS),(ICOM(12), MODE),
     +(ICOM(13),LNGTH),(ICOM(14),  KT1),(ICOM(15),  KT2),
     +(ICOM(16),  KT3),(ICOM(17), LFRQ),(ICOM(22),  KEY),
     +(ICOM(35), IFST),(ICOM(36), ILST),(ICOM(40), JDIR),
     +(ICOM(45),   IX),(ICOM(46),   IY),(ICOM(47),INCRX),
     +(ICOM(48), KTHR),(ICOM(51), IHDR),(ICOM(101),IDATE)

      DATA NQUAL/2048,100B,16383,45,0/

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 2 (ON)
      IF(ISSW(2) .LT. 0)       PAUSE 7
```

```

      LBEAM = -1

C - - - - - - INITIALIZE PLOT PARAMETERS
      IRESET = 37B
      IDASH  = 55B

C- - - - - - GET COMMON BLOCK DATA, STORE IN ICOM - - - -
      CALL PUTQ(0, NQUAL)
      CALL KYBD(2HMS, 31, 0)
      CALL KYBD(2HMS, 11, 0)
      DO 10  I = 1, 127
       CALL GETW(0, I, ICOM(I), II)
   10 CONTINUE
      IDOP0 = NEWBS / 4
      CALL PTIME(IFST,ICOM)

C - - - - - - - - - - - - GENERATE LIST OF DESIRED BEARINGS (NOMINAL) -
   15 BRG = JDIR + 78.75
      DO 20  J = 1, 8
       IF(BRG .LT.   0.)  BRG = BRG + 360
       IF(BRG .GE. 360.)  BRG = BRG - 360
       LISTBM(J) = IFIX(BRG + 0.5)
   20 BRG = BRG - 22.5

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 10 (ON)
      IF(ISSW(10) .LT. 0) WRITE(2, 920) ICOM
  920 FORMAT("MP ",10I6)

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 4 (ON)
      IF(ISSW(4) .LT. 0) WRITE(1, 921)LISTBM,KTHR,IFST, ILST
  921 FORMAT(20X, 8I4)

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 10 (ON)
       IF(ISSW(10) .LT. 0)  PAUSE 15
```

```
  100  CALL MPGRD(ICOM, LISTBM)

C  - - - - - - - - BEGIN PING LOOP - - - -
       DO 200  NP = IFST, ILST
        NPAGE = 0

C - - -  GET FIRST DATA BLOCK PAGE FOR THIS PING, SET LOCAL RETURN
        ASSIGN 105 TO LRET
                                       GO TO 300
  105   IF(MAT(1, 1) .GT. 360)         GO TO 200
        K = 1

C - - - - - BEGIN PROCESSING THIS DATA POINT LINE
  110   KBEAM = MAT(1, K)

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 9 (ON)
        IF(ISSW(9) .LT. 0) WRITE(1, 9320) (MAT(I,K), I = 1, 4)

        IF(KBEAM .EQ. LBEAM)        GO TO 115

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 10 (ON)
        IF(ISSW(10) .LT. 0) WRITE(1, 991) NP, NRBLK, LBEAM,KBEAM
  991    FORMAT(1X, 10I6)

C - - - IS THIS BEAM CLOSE TO A BEAM IN THE LIST FOR PLOTTING? - -
        LISTED = 1
        DO 112  NDX = 1, 8
         IDIF = IABS(LISTBM(NDX) - KBEAM)
         IF(IDIF .LE. 12 .OR. IDIF .GE. 348)  GO TO 114
   112  CONTINUE
        LISTED = 0
                    GO TO 150

C - - - - SET CURSOR Y LOCATION FOR THIS BEAM& PING, MARK THIS LIN
```

```
 114   LY = IY + (NDX-1)*80 + (NP-1)*9
       CALL TPLOT(0, IX,   LY+8)
       CALL TPLOT(1, IX-5, LY+4)
       CALL TPLOT(1, IX,   LY)
       CALL CHOUT(IRESET)
       LY = LY + 2
       LBEAM = KBEAM

C - - - - - - IF THIS BEAM IS ON THE LIST, GO TO CW OR FM PROCESSING
 115   IF(LISTED .EQ. 0)            GO TO 150
       IF(MODE .GE. 1)              GO TO 125

C  C.W. PROCESSING:  - - - - - -    GET VALUES FOR THIS DATA LINE
 116   KEXCED= MAT(3, K)
       IF(KEXCED .LT. KTHR)         GO TO 150
       KRANGE= MAT(2, K)
       MAXCED= KEXCED

       ASSIGN 122 TO LRET
 120   IF(K .GE. 512)       GO TO 300

C - - - - - - COMPARE NEXT DATA LINE BEAM AND RANGE VALUES WITH THIS
 122    IF(KBEAM .NE. MAT(1,K+1) .OR. KRANGE .NE. MAT(2,K+1))
     *                                    GO TO 130

C - - - - - - - - SAME.  SAVE MAX. EXCEEDENCE VALUE. - - -
        MAXCED = MAX0(MAXCED, MAT(3, K+1) )
        K = K + 1
                 GO TO 120
C < < < < < < < <<<<<<<<<<

C  F.M. PROCESSING:                TEST FOR ZERO DOPPLER - - -
 125   IDIF = IABS(MAT(4, K) - IDOP0)

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 6 (ON)
```

```
      IF(ISSW(6) .LT. 0) WRITE(1, 9125) (MAT(I, K), I = 1, 4),
     *                                                  IDOP0, IDIF
 9125    FORMAT(2X, 6I8)

C        IF(IABS(MAT(4, K) - IDOP0) .GT. 15)   GO TO 150

C TEST TEST TEST    ALL DOPPLERS OR'ED    TEST TEST

C - - - - - - DATA IS WITHIN 15 CELLS OF ZERO DOPPLER!  TEST THRESHO
      MAXCED = MAT(3, K)

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 6 (ON)
      IF(ISSW(6) .LT. 0) WRITE(1, 9125) KTHR, MAXCED, KRANGE, K

      IF(MAXCED .LT. KTHR)                          GO TO 150
      KRANGE = MAT(2, K)
       KTEXC = 1

C - - - - - - - - (((F.M. DE-SKEW ALGORITHM.  ACCUMULATE
C                    EXCEEDECES ON THIS DIAGONAL.))))) * * * *
C      MAXCED = KTEXC
C      IF(KTEXC .LT. 1)                             GO TO 150

C  - - - - - - CALCULATE AND PLOT COORDINATES FOR THIS POINT - - -
 130   ICELX = (KRANGE - MIN) * 890.0 / (MAX - MIN) + 127.5

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 9 (ON)
      IF(ISSW(9) .LT. 0) WRITE(2, 9130)K,MAXCED,MIN,MAX,INCRX,
     *ICELX,LY,KBEAM,KRANGE
 9130    FORMAT(9I6)

       IF(ICELX .LT. IX .OR. ICELX .GT. 1019) GO TO 150
       KVAL   = MIN0(10, IFIX(0.1*MAXCED + 0.5) )
       MY     = MIN0(780, LY + KVAL)
       IENDX  = MIN0(1019, ICELX + INCRX-4)
```

```
      CALL TPLOT(0, ICELX, LY)
      DO 140  LX = ICELX, IENDX, 4
       CALL TPLOT(1, LX+2, MY)
       CALL TPLOT(1, LX+4, LY)
 140  CONTINUE

C - - - - - - - INCREMENT DATA LINE INDEX, TEST FOR END OF DATA - -
 150  K = K + 1
      ASSIGN 105 TO LRET
      IF(K .GT. 512)            GO TO 300
      IF(MAT(1, K) .LT. 360)  GO TO 110

C - - - - - - - - END OF PING LOOP - - - - - -
      LBEAM = -1
 200  CONTINUE

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 2 (ON)
      IF(ISSW(2) .LT. 0)        PAUSE 200

      CALL TPLOT(0,0,0)
      CALL CHOUT(IRESET)

C - - - - - - - - OPTION TO CHANGE THRESHOLD AND REPLOT
      KTHR  = 0
      WRITE(1, 9200)
9200  FORMAT(/" ENTER NEW THRESHOLD PERCENTAGE FOR PLOT (1 - 100%)
     * "  (1% DISPLAYS ALL ANALYZED AMPLITUDES. 100%= NONE)"/
     * "  ENTER 0 TO RETURN TO ANALYSIS PROGRAM")
      CALL INPT(KTHR, 0, 100)
      IF(KTHR .GT. 0)           GO TO 15
                          RETURN
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

C  - - - - - - - GET DATA BLOCK FROM DISC  - - -
 300  NPAGE = NPAGE + 1
```

```
      IF(NPAGE .GT. 3)          GO TO 200
      NRBLK = 3 * (NP - 1) + NPAGE

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 10 (ON)
       IF(ISSW(10) .LT. 0) WRITE(1, 992)NRBLK
 992    FORMAT("READING BLOCK NO. ", I4,"   FROM DISC")

      CALL PUTQ(0, NQUAL)
      CALL KYBD(2HMS,31,NRBLK)
      CALL KYBD(2HMS,11,0)

C - - - - - - MOVE DATA FROM CORE BLOCK 0 TO MATRIX (4 X 512)
      DO 310  L = 1, 2048
       CALL GETW(0, L-1, MAT(L), II)
       IF(MAT(L) .NE. 999)      GO TO 310
       IF(MOD(L-1,4) .EQ.  0)   GO TO 320
 310  CONTINUE
      L = L - 1

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 2 (ON)
 320  IF(ISSW(2) .LT. 0) PAUSE 320

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 8 (ON)
      IF(ISSW(8) .LT. 0) WRITE(2, 9320) (MAT(J), J = 1, L)
9320  FORMAT(10X, 4I8)

C - - - - - - RESET LINE COUNTER, LOCAL RETURN TO 105 OR 122
      K     = 0
                 GO TO LRET
C  *************  END OF MULTI-PING DISPLAY ROUTINE  *************
      END

      SUBROUTINE MPGRD(ICOM, LISTBM)
C                                   DEC. 29, 1977
C  ******************************************************************
```

```
C   *  MULTI-PING PLOT GRID GENERATION ROUTINE                         *
C   ****************************************************************

        DIMENSION LISTBM(8), ICOM(127),LABEL(2), IDATE(9)

             IF (ISSW(14) .LT. 0) GO TO 208
         THRES = ICOM( 5) * 1E-6
  208    KODE = ICOM( 7)
         MODE = ICOM(12)
         LNGTH= ICOM(13)
         KT1  = ICOM(14)
         KT2  = ICOM(15)
         KT3  = ICOM(16)
         LFRQ = ICOM(17)
         IFST = ICOM(35)
         ILST = ICOM(36)
         IDPLR= ICOM(37)
         JSTRT= ICOM(38)
         JEND = ICOM(39)
         JDIR = ICOM(40)
         KTHR = ICOM(48)
         DO 10  I = 1, 9
  10     IDATE(I) = ICOM(I+100)

C - - - - - - - - INITIALIZE PLOT PARAMETERS / ERASE SCREEN. - -
        IRESET = 37B
        IESCAP = 33B
        IERASE = 14B
        CALL CHOUT(IESCAP)
        CALL CHOUT(IERASE)
        CALL D750

C - - - - - - - - COMPUTE HORIZONTAL SCALING - - -
        DELTI  = 1.0 * JEND - JSTRT
```

```
C - - - - - -  LOCATE INITIAL ORIGIN IN X AND Y - - -
      NSEPX = 112
      NSEPY =  80
      IX = 1023 - (8*NSEPX)
      IY =  780 - (8*NSEPY)

C - - - - - - CONSTRUCT VERTICAL GRID - - -
      NY = IY - 30
      DO 20  J = 0, 8
       LX = IX + J * NSEPX
       CALL TPLOT(0, LX, IY)
       CALL TPLOT(1, LX,780)

C - - - -GET COORDINATES FOR NUMBERS, LABEL X-AXIS - -
       TYARDS = JSTRT + J * DELTI/8.0
       NX = LX - 50
       IF(J .GE. 8)   NX = LX - 70
       CALL TPLOT(0, NX, NY)
       CALL CHOUT(IRESET)
       WRITE(2, 920) TYARDS
 920   FORMAT(F5.1)
  20  CONTINUE

C - - - - - - - - CONSTRUCT HORIZONTAL GRID
      NX = IX - 70
      DO 30  J = 1, 9
       LY = IY + (J-1) * NSEPY
       CALL TPLOT(0,   IX, LY)
       CALL TPLOT(1, 1023, LY)

C - - GET COORD. FOR BEAM NUMBERS, LABEL Y-AXIS AREAS - - -
       IF(J .GT. 8)              GO TO 30
       IF(LISTBM(J) .EQ. 0)      GO TO 30
       NY = LY + 30
       CALL TPLOT(0, NX, NY)
```

```
            CALL CHOUT(IRESET)
            WRITE(2, 930) LISTBM(J)
  930       FORMAT(I4)
   30     CONTINUE

C - - - - - - - - WRITE HEADER DATA FOR PLOT - - -
        CALL TPLOT(0, 1, 80)
        CALL CHOUT(IRESET)
                            LABEL(1) = 2HHA
                            LABEL(2) = 2HLF
        IF(IDPLR .EQ. 1) LABEL(1) = 2HFU
        IF(IDPLR .EQ. 1) LABEL(2) = 2HLL
        WRITE(1, 940)IFST,LABEL(1),LABEL(2),KTHR
  940   FORMAT("FIRST PING NO:",I2, 5X,"DOPPLER: ",2A2,6X,
       + "PLOT THRESHOLD:",I3)
                            LEN  = 2HT2
        IF(LNGTH .GE. 1) LEN  = 2HT1
                            LFRE = 2HF2
        IF(LFRQ  .EQ. 1) LFRE = 2HF1
                            LTYP = 2HCW
        IF(MODE .GE. 1)  LTYP = 2HFM
        WRITE(1, 941)LTYP,LEN,LFRE,KODE,KT1,KT2,KT3,IDATE
  941   FORMAT("SIGNAL:",2X,A2,2X,A2,2X,A2,2X,,"HANNING:",I2,
       +9X,"TIME: ",I2,":",I2,":",I2,3X,9A2)

C - - - - - SAVE PLOT ORIGIN
          ICOM(45) = IX
          ICOM(46) = IY
                      RETURN
C  ############  END OF MULTI-PING PLOT GRID ROUTINE  ############
        END

        SUBROUTINE FPGRD(ICOM)
C  *************************************************************
C  *  FOUR PART DISPLAY GRID GENERATION ROUTINE                *
```

```
C   ***************************************************************

       DIMENSION ICOM(127), IX(4), IY(4)

C - - - - - - - - INITIALIZE TABLE OF X AND Y LIMITS OF FOUR GRIDS
       DATA IX/60,500,570,1010/,IY/80,410,450,780/
       DATA ISEPX/110/,ISEPY/55/

       JSTRT = ICOM(38)
       JEND  = ICOM(39)

C - - - - - - - - - - - - - - INITIALIZE PLOT PARAMETERS, ERASE SCREEN
       IRESET = 37B
       IESCAP = 33B
       IERASE = 14B
       CALL CHOUT(IESCAP)
       CALL CHOUT(IERASE)
       CALL D750

C   - - - - - - - - - - CONSTRUCT VERTICAL GRID
       DO 20  I = 0, 4
        TYARDS = JSTRT + I * (JEND - JSTRT)/4.0

C >>>>>>> DIAGNOSTIC >> SENSE SWITCH 10 (ON)
        IF(ISSW(10) .LT. 0)WRITE(1, 910) TYARDS, JSTRT, JEND
 910     FORMAT(" TYARDS", F10.1, 2I6)

        DO 20 J = 1, 3, 2
         LX = IX(J) + I * ISEPX
         CALL TPLOT(0, LX, IY(1) )
         CALL TPLOT(1, LX, IY(2) )
         CALL TPLOT(0, LX, IY(3) )
         CALL TPLOT(1, LX, IY(4) )

C - - - - - - - - - - LABEL X AXIS
```

```
                       MOVEH = 45
            IF(I .EQ. 4) MOVEH = 55
            CALL TPLOT(0, LX-MOVEH, IY(1) - 25)
            CALL CHOUT(IRESET)
            WRITE(1, 920) TYARDS
            CALL TPLOT(0, LX-MOVEH, IY(3) - 25)
            CALL CHOUT(IRESET)
            WRITE(1, 920) TYARDS
   20     CONTINUE
  920       FORMAT(F5.1)

C - - - - - - - - - CONSTRUCT HORIZONTAL GRIDS
          DO  30 J = 0, 6
           YDOP = -15 + 5 * J
           YBRG = 360 - J*60
           DO 30 I = 1, 3, 2
            LY = IY(I) + J * ISEPY
            CALL TPLOT(0, IX(1), LY)
            CALL TPLOT(1, IX(2), LY)
            CALL TPLOT(0, IX(3), LY)
            CALL TPLOT(1, IX(4), LY)

C - - - - - - - - - - - - - LABEL Y-AXIS
            CALL TPLOT(0, IX(3)-45, LY-8)
            CALL CHOUT(IRESET)
            WRITE(1, 930) YDOP
            CALL TPLOT(0, IX(1)-45, LY-8)
            CALL CHOUT(IRESET)
            IF(I .EQ. 1)  WRITE(1, 930) YBRG
            IF(I .EQ. 3)  WRITE(1, 930) YDOP
   30     CONTINUE
  930       FORMAT(I3)
C  ######  END OF FOUR PART DISPLAY GRID GENERATION ROUTINE  #####
          END
```

```
      SUBROUTINE PLLIN (IX,INCRX,IY,IDTA,MAX)
C ****************************************************************
C *  THIS SUBROUTINE PLOTS A VERTICAL BAR WITH BASE AT IX, IY.   *
C *  HEIGHT IS 2*IDTA.                                            *
C ****************************************************************

      DATA  IRESET/37B/

      IF(IDTA .GT. 10) IDTA = 10
      MY    = MIN0(MAX, IY + IDTA + IDTA)
      INCR = INCRX * 0.4991 + 0.5
      IENDX = MIN0(1019, IX + INCR - 4)
      CALL TPLOT(0, IX, IY)
      DO 15  LX = IX, IENDX, 4
       CALL TPLOT(1, LX+2, MY)
       CALL TPLOT(1, LX+4, IY)
 15   CONTINUE
      CALL CHOUT(IRESET)
C ###############  END OF THE LINE PLOT ROUTINE  ###############
      END

      SUBROUTINE BRDB
C ****************************************************************
C *  DUMP DATA BLOCK SUBROUTINE                                   *
C ****************************************************************

      DIMENSION IBUF(20), KQUAL(5), NQUAL(5)

      DATA NQUAL/2048,100B,16383,45,0/

      DATA NDBLK/0/,KBLK/0/,KOLS/10/,LINES/100/
      KTDAT = 0
      WRITE(1, 910)
 910  FORMAT(" ENTER DISC BLOCK NO. (-1 IF NONE)")
      CALL INPT(NDBLK, -1, 33)
```

```
      WRITE(1, 911)
911   FORMAT(" ENTER CORE BLOCK NUMBER (0 - 3)")
      CALL INPT(KBLK, 0, 3)
      WRITE(1, 912)
912   FORMAT(" NO. OF COLUMNS FOR PRINTOUT (1 - 10)")
      CALL INPT(KOLS, 1, 10)
      WRITE(1, 913)
913   FORMAT(" ENTER NO. OF LINES FOR PRINTOUT")
      CALL INPT(LINES, 1, 2048)
      IF(NDBLK .LT. 0) GO TO 10

C - - - - - - - - - - - - - - - -   DISC BLOCK INTO CORE
      CALL GETQ(KBLK, KQUAL)
      CALL PUTQ(KBLK, NQUAL)
      CALL KYBD(2HMS, 31, NDBLK)
      CALL KYBD(2HMS, 11, KBLK)
      CALL ENDIO

C - - - - - - - - - - - - - - - -   WRITE DATA FROM CORE DATA BLOCK
 10   DO  30 L = 0, LINES-1
       DO 20 K = 0, KOLS -1
        KTDAT = KTDAT + 1
        NWORD = K + L * KOLS
        CALL GETW(KBLK, NWORD, IBUF(K+1), II)
 20    CONTINUE
       LK = L+1
       WRITE(1, 930)LK, (IBUF(I), I = 1, KOLS)
 30   CONTINUE
930    FORMAT(1X, 11I6)
      CALL ENDIO
      CALL PUTQ(KBLK, KQUAL)

C > > > > RETURN TO CALLER
        RETURN
C  ############  END OF DUMP DATA BLOCK SUBROUTINE  ##############
```

```
      END

      SUBROUTINE PTIME(IPNG, ICOM)

C **************************************************************
C *  THE PTIME SUBROUTINE CONVERTS PING TIME FROM BCD UNITS    *
C *  (OBTAINED FROM THE HEADER WRITTEN DURING DATA COLLECTION) *
C *  INTO BINARY FORM.   RESULTS ARE STORED IN THE PROPER ICOM *
C *  LOCATIONS.                                                *
C **************************************************************

      DIMENSION ICOM(127), MASK(4), IBCD(6), IHDR(16)

      DATA      MASK/17B, 360B, 7400B, 170000B/

C > > > > GET PING TIME FROM HEADER
      CALL HEADR (IPNG, IHDR)

C > > > > COMPUTE HOURS
      IBCD(5)  = IAND (MASK(1), IHDR(1))
      IBCD(6)  = IAND (MASK(2), IHDR(1))
      ICOM(14) = IBCD(6) / 20B * 10 + IBCD(5)

C > > > > COMPUTE MINUTES AND SECONDS
      DO 10 I = 1,4
   10 IBCD(I) = IAND (MASK(I), IHDR(2))

      ICOM(15) = IBCD(4) / 10000B * 10 + IBCD(3) / 400B
      ICOM(16) = IBCD(2) /    20B * 10 + IBCD(1)

C > > > > RETURN TO CALLER
      RETURN

C ****************  END OF PING TIME SUBROUTINE  ****************
```

```
C   ********************************************************************
C   ***************  END OF PLOT DISPLAY SUBROUTINE  ****************
C   ********************************************************************
      END
      END$

:
```

## 5.3 OLAAS PROCESSING & DISPLAY SYSTEM SUBROUTINES

The OLAAS System subroutines are used in conjunction with the major functional programs/subroutines as part of the modular concept used to design the system.

The following descriptions, flow diagrams, and listings are the subroutines used in the system:

ASKQ Subroutine
CDUMP Subroutine
CLOAD Subroutine
DMA1 Subroutine
D75Ø Subroutine
INPT Subroutine
INPUT Subroutine
ISHFT Subroutine
LINK6 Subroutine
LKTBG Subroutine
RDATA Subroutine
TAT1 Subroutine
TEK Subroutine
TYPI Subroutine
USTRT Subroutine

QUESTIONS/ANSWERS FOR Y1ØØ2 SUBROUTINE (ASKQ)

1. FUNCTION. The Question/Answer(ASKQ) subroutine is used to place the program operation parameter question on the screen with the previous parameter values given. User enters new parameters or uses the previous parameter values. Data entered is checked for range errors.

2. CONSTRAINTS. All input variables must be integers. This subroutine is used only with the Y1ØØ2.

3. CALLING SEQUENCE. Called by: Y1ØØ2.

Call ASKQ (IVAR, IMIN, IMAX, IQUE), IDATE

IVAR = Variable

IMIN = Minimum value allowed

IMAX = Maximum value allowed

IQUE = Question number

IDATE = Date

4. DESCRIPTION OF INPUT. Input is a typed value as requested by the displayed question.

5. DESCRIPTION OF OUTPUT. Output is the variables described in the calling sequence, in integer format. Printed output is the operator prompts.

6. FILES USED. N/A.

7. ERROR AND RESTRICTIONS. A message is output to the screen indicating an "out of range" value was entered. The subroutine then waits for a correct input value.

8. COMPUTER OPERATOR INSTRUCTIONS. When the ASKQ subroutine is called during normal system operation, the operator is prompted by a question appearing on the screen. The operator replies by entering a value through the keyboard and pressing a carriage return.

9. DESCRIPTION OF PROCESSING. ASKQ subroutine at (1) initializes the printer buffer length variable, IBUFL, to the number of characters to be output in each question. The print buffer area, IPBUF, is cleared each time the subroutine is entered.

At (2), determine the question entry point base on the question number parameter passed from the calling subroutine, Y1ØØ2. Each of the thirteen (13) questions are handled in the same manner.

i.e., A call to the Fortran IV library routine, CODE, causes an internal conversion in the Fortran Formatter and allows the screen's cursor to remain at the end of the displayed message. The requested question is set-up and control is transferred to (3).

At (3), the Print Question, PNTL, subroutine is called to output the question to the screen. Note that the last entered value or the initialized value of the selected question is displayed with the question.

At (4), the keyboard input subroutine, TYPI, is called to allow the User Operator to enter an answer to the question. The three (3) possible answers are as follows:

A. Carriage Return = Use the same value now displayed on the screen (last entered value).

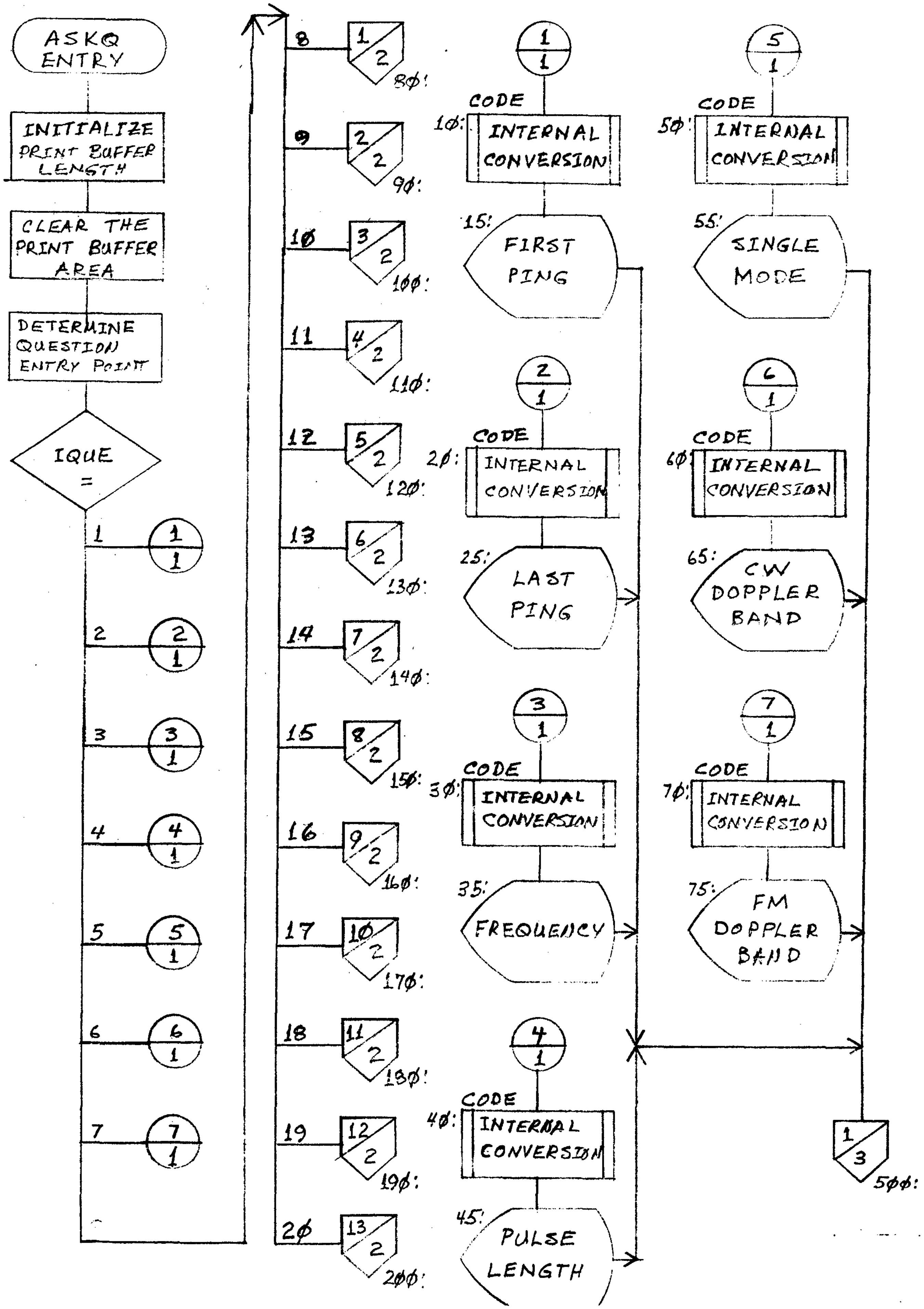
B. 12345 = Abort the operation and return to Coreload Ø, OLAAS Executive Program.

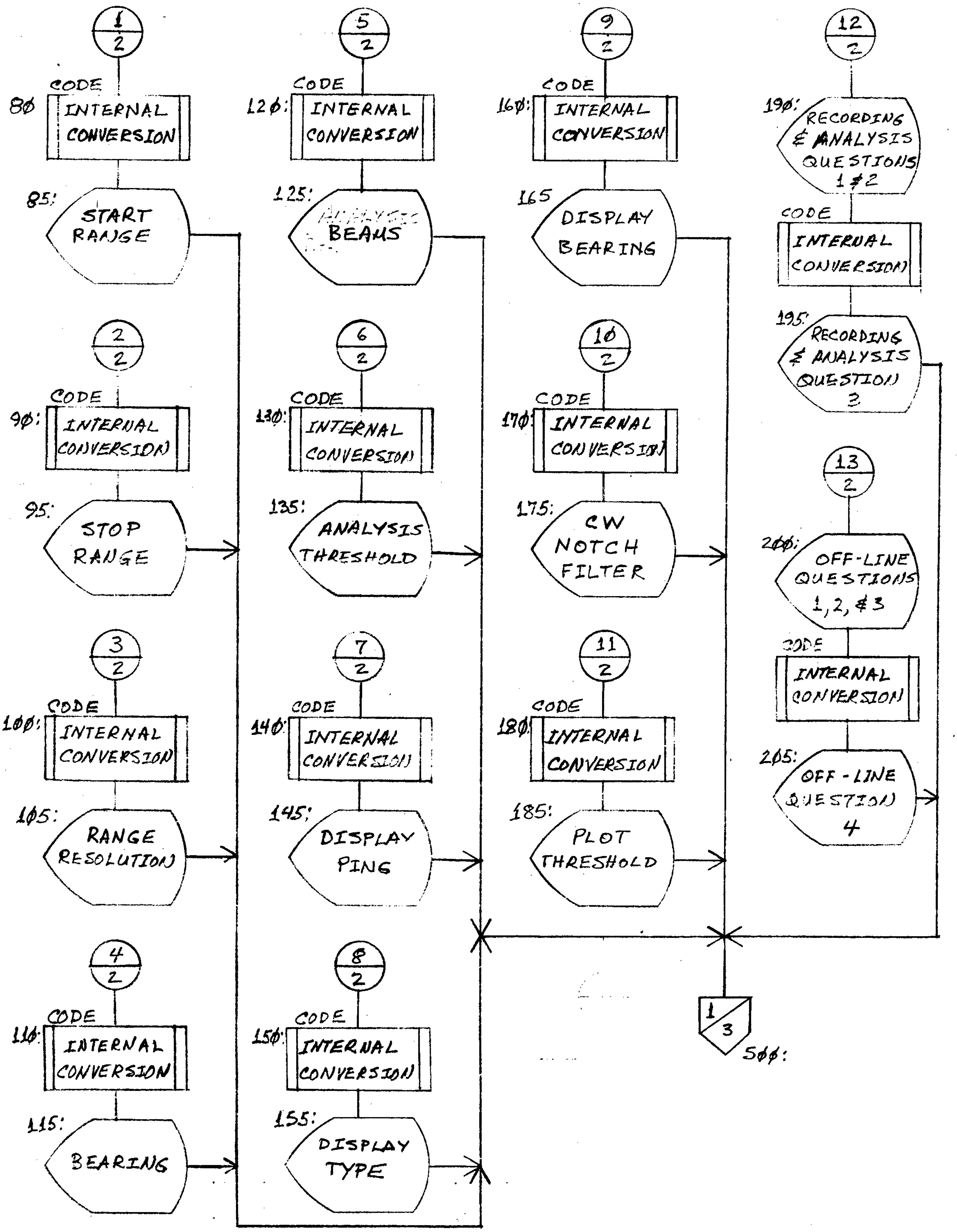
C. Enter new value = This is the new parameter that is to be used in the calling program.

At (5), the new value entered is checked for an "out-of-range" error condition. If the value is out of range a message is displayed on the screen and control is passed back to output the question again and wait for the next input. The subroutine will repeat in the loop until a good value is entered.

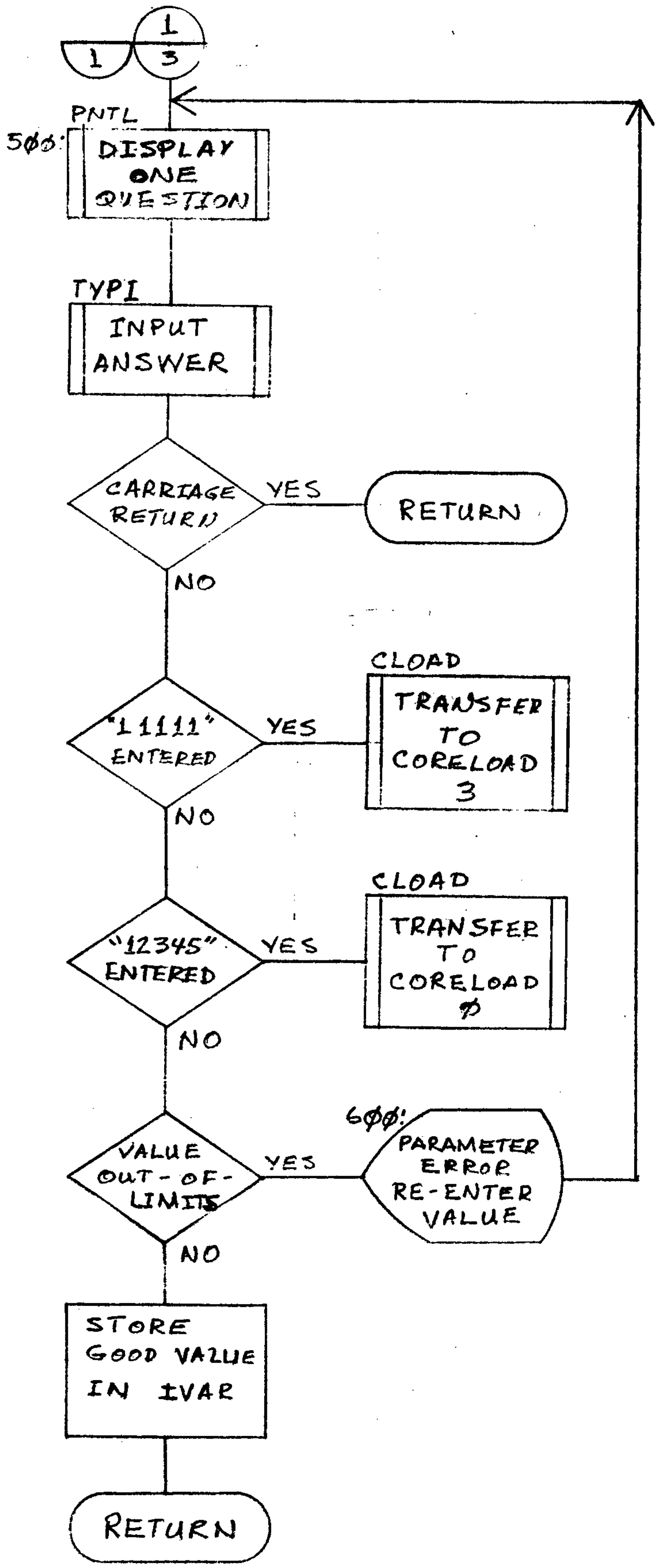
At (6), the new value is loaded into the variable parameter, IVAR, and the ASKQ subroutine exits returning control back to the calling subroutine, Y1ØØ2.

# ANALYSIS QUESTION/ANSWER SUBROUTINE (ASKQ)





1/2
CODE
80 INTERNAL CONVERSION
85: START RANGE
2/2
CODE
90: INTERNAL CONVERSION
95: STOP RANGE
3/2
CODE
100: INTERNAL CONVERSION
105: RANGE RESOLUTION
4/2
CODE
110: INTERNAL CONVERSION
115: BEARING
5/2
CODE
120: INTERNAL CONVERSION
125: BEAMS
6/2
CODE
130: INTERNAL CONVERSION
135: ANALYSIS THRESHOLD
7/2
CODE
140: INTERNAL CONVERSION
145: DISPLAY PING
8/2
CODE
150: INTERNAL CONVERSION
155: DISPLAY TYPE
9/2
CODE
160: INTERNAL CONVERSION
165 DISPLAY BEARING
10/2
CODE
170: INTERNAL CONVERSION
175: CW NOTCH FILTER
11/2
CODE
180: INTERNAL CONVERSION
185: PLOT THRESHOLD
12/2
190: RECORDING & ANALYSIS QUESTIONS 1 & 2
CODE
INTERNAL CONVERSION
195: RECORDING & ANALYSIS QUESTION 3
13/2
200: OFF-LINE QUESTIONS 1, 2, & 3
CODE
INTERNAL CONVERSION
205: OFF-LINE QUESTION 4
1/3
500:


1
1
3
PNTL
5ØØ:
DISPLAY ONE QUESTION
TYPI
INPUT ANSWER
CARRIAGE RETURN
YES
RETURN
NO
CLOAD
"11111" ENTERED
YES
TRANSFER TO CORELOAD 3
NO
CLOAD
"12345" ENTERED
YES
TRANSFER TO CORELOAD Ø
NO
6ØØ:
VALUE OUT-OF-LIMITS
YES
PARAMETER ERROR RE-ENTER VALUE
NO
STORE GOOD VALUE IN IVAR
RETURN

```
ASKQ   T=00004 IS ON CP00001 USING 00039 BLKS R=0160

FTN4
      SUBROUTINE ASKQ (IVAR,IMIN,IMAX,IQUE,IDATE)

C   *************************************************************
C   *                                                           *
C   *  TITLE:  Y1002 SUBROUTINE QUESTION/ANSWER PROGRAM         *
C   *                                                           *
C   *  MODULE:  ASKQ       SOURCE:  ASKQ       VERSION:  X01    *
C   *                      OBJECT:  #ASKQ                       *
C   *                                                           *
C   *  AUTHOR:  W. BOWEN          DATE:  25 JULY 1977           *
C   *                                                           *
C   *  MODIFICATIONS:                                           *
C   *    1:   ADD DISPLAY QUESTIONS                             *
C   *         (INDIVIDUAL BEAM, MULTI-PING, & FOUR PART DISPLAY) *
C   *           12/13/77     W. BOWEN                           *
C   *                                                           *
C   *  MODULE FUNCTION:                                         *
C   *     THE ASK QUESTION (ASKQ) SUBROUTINE IS USED TO PLACE   *
C   *  THE PROGRAM OPERATION PARAMETER QUESTIONS ON THE SCREEN  *
C   *  THE PREVIOUS PARAMETER VALUES GIVEN.   USER ENTERS NEW   *
C   *  PARAMETERS OR USES THE PREVIOUS PARAMETER VALUES.        *
C   *  ENTRY DATA IS CHECKED FOR RANGE ERRORS.                  *
C   *                                                           *
C   *  CALLING SEQUENCE:    CALLED BY - Y1002                   *
C   *                                                           *
C   *      CALL ASKQ (IVAR,IMIN,IMAX,IQUE)                      *
C   *                                                           *
C   *  INPUT PARAMETERS:    IVAR = VARIABLE                     *
C   *                       IMIN = MINIMUM VALUE ALLOWED        *
C   *                       IMAX = MAXIMUM VALUE ALLOWED        *
C   *                       IQUE = QUESTION NUMBER              *
C   *
```

```
C   *                              IDATE = DATE ENTERED                    *
C   *                                                                      *
C   *  OUTPUT PARAMETERS:   IVAR = VARIABLE                                *
C   *                                                                      *
C   ************************************************************************
C
C   *********************************
C   *  DEFINE ARRAYS AND VARIABLES  *
C   *********************************
C
       DIMENSION IPBUF(32), IDATE(9)
C
       IBUFL = 64
C
C   ***************************
C   *  INITIALIZE SUBROUTINE  *
C   ***************************
C
C > > > > CLEAR THE PRINT BUFFER AREA
       DO 5 I=1,32
       IPBUF(I) = 020040B
     5 CONTINUE
C > > > > DETERMINE QUESTION ENTRY POINT
       GO TO (10,20,30,40,50,60,70,80,90,100,110,120,130,
      +140,150,160,170,180,190,200) IQUE
C
C   ********************************************
C   *  ANALYSIS & DISPLAY PARAMETER QUESTIONS  *
C   ********************************************
C
C > > > > NUMBER OF PINGS
    10 CALL CODE
       WRITE  (IPBUF,15) IVAR
    15 FORMAT ("FIRST PING",9X,"NUMBER FOR ANALYSIS (1 TO 10)",
      +7X,"(",I5,")",2X)
```

```
      GO TO 500
C
   20 CALL CODE
      WRITE  (IPBUF,25) IVAR
   25 FORMAT ("LAST PING",10X,"NUMBER FOR ANALYSIS (1 TO 10)",
     +7X,"(",I5,")",2X)
      GO TO 500
C > > > > FREQUENCY (F1 OR F2)
   30 CALL CODE
      WRITE  (IPBUF,35) IVAR
   35 FORMAT ("FREQUENCY",10X,"F1 = 1  OR  F2 = 2",18X,
     +"(",I5,")",2X)
      GO TO 500
C > > > > PULSE LENGTH  (SHORT OR LONG)
   40 CALL CODE
      WRITE  (IPBUF,45) IVAR
   45 FORMAT ("PULSE LENGTH",7X,"SHORT = 2  OR  LONG = 5",13X,
     +"(",I5,")",2X)
      GO TO 500
C > > > > SIGNAL MODE  (CW, FM UP OR FM DOWN)
   50 CALL CODE
      WRITE  (IPBUF,55) IVAR
   55 FORMAT ("SIGNAL MODE",8X,"CW = 0, FM UP = 1, OR FM DOWN"
     +" = 2",3X,"(",I5,")",2X)
      GO TO 500
C > > > > CW DOPPLER BAND  (FULL, HALF, UP, OR DOWN)
   60 CALL CODE
      WRITE  (IPBUF,65) IVAR
   65 FORMAT ("CW DOPPLER BAND",4X,"FULL=1, HALF=2, UP=3, "
     +"OR DOWN=4",5X,"(",I5,")",2X)
      GO TO 500
C > > > > FM DOPPLER BAND  (FULL OR HALF)
   70 CALL CODE
      WRITE  (IPBUF,75) IVAR
   75 FORMAT ("FM DOPPLER BAND",4X,"FULL = 1  OR  HALF = 2",
```

```
      +14X,"(",I5,")",2X)
       GO TO 500
C > > > > START RANGE
    80 CALL CODE
       WRITE  (IPBUF,85) IVAR
    85 FORMAT ("START RANGE",8X,"IN MULTIPLES OF 10 KILOYARDS",
      +8X,"(",I5,")",2X)
       GO TO 500
C > > > > STOP RANGE
    90 CALL CODE
       WRITE  (IPBUF,95) IVAR
    95 FORMAT ("STOP RANGE",9X,"IN MULTIPLES OF 10 KILOYARDS",
      +8X,"(",I5,")",2X)
       GO TO 500
C > > > > RANGE RESOULTION
   100 CALL CODE
       WRITE  (IPBUF,105) IVAR
   105 FORMAT ("RANGE RESOLUTION",3X,"IN MULTIPLES OF 160 YARDS",
      +11X,"(",I5,")",2X)
       GO TO 500
C > > > > BEARING
   110 CALL CODE
       WRITE  (IPBUF,115) IVAR
   115 FORMAT ("BEARING",12X,"ANGLE IN DEGREES (CENTER BEAM)",
      +6X,"(",I5,")",2X)
       GO TO 500
C > > > > BEAMS
   120 CALL CODE
       WRITE  (IPBUF,125) IVAR
   125 FORMAT ("BEAMS",14X,"NUMBER OF BEAMS",21X,"(",I5,")",2X)
       GO TO 500
C > > > > THRESHOLD
   130 CALL CODE
       WRITE  (IPBUF,135) IVAR
   135 FORMAT ("ANALYSIS THRESHOLD",1X,"0 TO 10000"
```

```
     +" (LEVEL IN MICRO-VOLTS)",3X,"(",I5,")",2X)
      GO TO 500
C > > > > DISPLAY PING NUMBER  (INDIVIDUAL BEAM DISPLAY)
  140 CALL CODE
      WRITE  (IPBUF,145) IVAR
  145 FORMAT (/"DISPLAY PING",7X,"PING NUMBER TO BE DISPLAYED",
     +9X,"(",I5,")",2X)
      GO TO 500
C > > > > DISPLAY TYPE  (INDIVIDUAL BEAM DISPLAY)
  150 CALL CODE
      WRITE  (IPBUF,155) IVAR
  155 FORMAT ("DISPLAY TYPE",7X,"2 = INDIVIDUAL   5 = BEAMS ",
     +"OR'ED",4X,"(",I5,")",2X)
      GO TO 500
C > > > > DISPLAY BEARING  (FOUR PART DISPLAY)
  160 CALL CODE
      WRITE  (IPBUF,165) IVAR
  165 FORMAT ("DISPLAY BEARING",4X,"BEARING TO BE DISPLAYED ",
     +"(0 TO 359)",2X,"(",I5,")",2X)
      GO TO 500
C > > > > CW NOTCH FILTER
  170 CALL CODE
      WRITE  (IPBUF,175) IVAR
  175 FORMAT ("CW NOTCH FILTER",4X,"WIDTH IN FREQUENCY CELLS"
     +"  (0 TO 25)",1X,"(",I5,")",2X)
      GO TO 500
C > > > > PLOT THRESHOLD
  180 CALL CODE
      WRITE  (IPBUF,185) IVAR
  185 FORMAT ("PLOT THRESHOLD",5X,"1 TO 100 (%)  (1= DISPLAY ALL)"
     *6X,"(",I5,")",2X)
      GO TO 500
C > > > > RECORDING & ANALYSIS QUESTIONS
  190 WRITE  (2,193) IDATE
  193 FORMAT (/"*****  OLAAS DATA RECORDING & ANALYSIS  "
```

```
      +"*****"/12X,9A2//5X,"** ENTER DESIRED FUNCTION **"/
      +"1 = ON-LINE DATA RECORDING & ANALYSIS"/
      +"2 = OFF-LINE ANALYSIS & PLOT DISPLAY"/
      +"3 = INITIALIZE CORELOAD 3 ONLY"/
      +"4 = OPERATOR: SET SWITCH 1 (ON) TO ASK THESE"
      +" THESE QUESTIONS:"/4X,"BUOY FREQUENCY, PULSE LENGTH,"
      +" STOP RANGE, & CW NOTCH FILTER")
       CALL CODE
       WRITE  (IPBUF,195) IVAR
   195 FORMAT ("5 = DATE ENTRY",41X,"(",I5,")",2X)
       GO TO 500
C > > > > OFF-LINE QUESTIONS
   200 WRITE  (2,203) IDATE
   203 FORMAT ("***** OFF-LINE DATA ANALYSIS & PLOT DISPLAY"
      +" *****"/16X,9A2//5X,"** SELECT OFF-LINE OPTION **"/
      +"1 = ANALYSIS OF PREVIOUSLY RECORDED DATA"/
      +"2 = INDIVIDUAL BEAM DISPLAY"/
      +"3 = MULTI-PING DISPLAY")
       CALL CODE
       WRITE  (IPBUF,205) IVAR
   205 FORMAT ("4 = FOUR PART DISPLAY",34X,"(",I5,")",2X)
       GO TO 500
C
C   **********************************************************
C   *  CALL PRINT LINE (PNTL) SUBROUTINE TO PRINT QUESTIONS  *
C   **********************************************************
C
   500 CALL PNTL (IPBUF,IBUFL)
C
C   **********************************************
C   *  CALL TO INPUT PARAMETERS FROM THE KEYBOARD  *
C   **********************************************
C
       CALL TYPI (IN,I1)
C > > > > RETURN IF CARRIAGE RETURN ONLY
```

```
      IF (I1 .EQ. 0) RETURN
C > > > > REPEAT THE LAST QUESTION ASKED  (IN = 11111)
      IF (IN .EQ. 11111) GO TO 500
C > > > > CHECK FOR RETURN TO EXECUTIVE  (IN = 12345)
      IF (IN .EQ. 12345) CALL CLOAD (1,2)
C > > > > CHECK IF DATA IS OUT OF RANGE  (IMIN TO IMAX)
      IF (I1 .LT. 0 .OR. IN .GT. IMAX .OR. IN .LT. IMIN)
     +GO TO 600
C
C   ****************************************
C   *  STORE GOOD VALUE IN IVAR AND RETURN  *
C   ****************************************
C
      IVAR = IN
      RETURN
C
C   *********************************
C   *  ERROR DETECTED, GET NEW INPUT  *
C   *********************************
C
  600 WRITE  (2,605)
  605 FORMAT ("********** PARAMETER ERROR **********      RE-"
     +"ENTER VALUE")
      GO TO 500
C
C  >>>>>>> END OF SUBROUTINE ASKQ >>>>>>>
      END
      END$
```

:

CORE DUMP ROUTINE - CDUMP

1. FUNCTION. CDUMP dumps all of core (locations 2-77577) onto six consecutive disc tracks specified in the calling sequence.

2. CONSTRAINTS. NIBBL must be in core in locations 77600-77677. It must be the unmodified HP version, as it is modified by CDUMP.

3. CALLING SEQUENCE. Called by: Y0001, Y0002

```
CALL CDUMP (ICYL)
```

where ICYL is the starting cylinder number of the desired coreload.

4. DESCRIPTION OF INPUT. None.

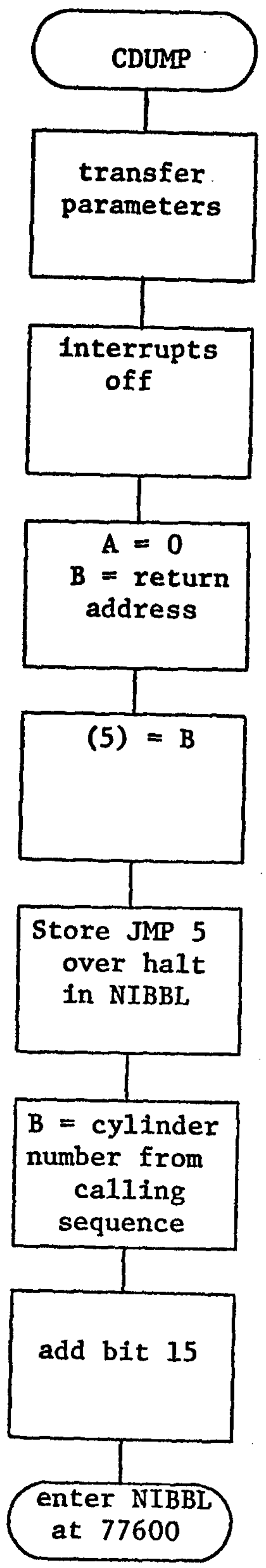
5. DESCRIPTION OF OUTPUT. Unknown.

6. FILES USED. N/A

7. ERRORS AND RESTRICTIONS. An irrecoverable error halt will occur if a disc transfer error or a disc status error occurs.

8. COMPUTER OPERATOR INSTRUCTIONS. N/A

9. DESCRIPTION OF PROCESSING. CDUMP uses NIBBL, the Disc Basic Binary Loader which reads or writes an entire coreload. However, NIBBL is a manually operated program and normally halts upon completion rather than return to a calling program. Therefore, CDUMP modifies NIBBL to cause it to return. Processing is as follows: CDUMP turns off interrupts, picks up CDUMP's return address, stores it in address 5, and stores a jump to location 5 in NIBBL where the completion halt usually is. As there are no unused locations in NIBBL in which to store an off-page pointer, a base page pointer must be used. Location 5 is a convenient location as it is reserved for parity error and memory protect interrupts, neither of which is used in this system. The cylinder number is picked up from the calling sequence, placed in the B register, bit 15 is set, and NIBBL is entered, with A=0. NIBBL will dump all of core up through address 77577 and return to the address in 5.


CDUMP
transfer parameters
interrupts off
A = 0
B = return address
(5) = B
Store JMP 5 over halt in NIBBL
B = cylinder number from calling sequence
add bit 15
enter NIBBL at 77600

```
0001                          ASMB,R,L
0003* THIS SUBROUTINE DUMPS THE 32K CORE LOAD TO THE CYLINDER
0004* SPECIFIED.  IT WORKS BY REPLACING THE HALT IN NIBBL WITH A
0005* RETURN AND THEN JUMPING TO NIBBL.  THE RETURN IS IN LOCATION
0006* 5 AND THEREFORE THE HALT ON PARITY ERROR IS DISABLED.
0007* NIBBL MUST BE IN CORE FOR THIS TO WORK.  IT IS NORMALLY BROUGHT
0008* IN BY THE BOOT AT 77750.
0009*
0010* CALLING SEQUENCE:     JSB CDUMP
0011*                       DEF *+2
0012*                       DEF CYLNO
0013*                          -OR-
0014*                    CALL CDUMP (CYLNO)
0015*
0016  00000                       NAM CDUMP
0017                              ENT CDUMP
0018                              EXT .ENTR
0019  00000 000000  CYLNO NOP
0020  00001 000000  CDUMP NOP
0021  00002 016001X       JSB .ENTR
0022  00003 000000R       DEF CYLNO
0023  00004 103100        CLF 0
0024  00005 002400        CLA
0025  00006 066001R       LDB CDUMP
0026  00007 074005        STB 5
0027  00010 066015R       LDB JMP5
0028  00011 176017R       STB HLTAD,I
0029  00012 166000R       LDB CYLNO,I
```

```
00013 046020R         ADB BIT15
00014 126016R         JMP NIBBL,I
00015 124005   JMP5   JMP 5,I
00016 077600   NIBBL  OCT 77600
00017 077657   HLTAD  OCT 77657
00020 100000   BIT15  OCT 100000
                      END
**  NO ERRORS *TOTAL **RTE ASMB 750420**
```

```
/ASMB: $END
:
```

CORE LOAD ROUTINE - CLOAD

1. FUNCTION. CLOAD loads all of core (locations 2-77577) from six consecutive disc tracks specified in the calling sequence, and transfers control to a location in the new core load specified in the calling sequence.

2. CONSTRAINTS. NIBBL must be in core in locations 77600-77677. It must be the unmodified HP version, as it is modified by CLOAD. The transfer address must be in base page.

3. CALLING SEQUENCE. Called by: Y0001, Y0002, Y0003

CALL CLOAD (ICYL, IXFER)

where ICYL is the starting cylinder number of the desired coreload
IXFER is the transfer address in the new coreload.

4. DESCRIPTION OF INPUT. Unknown.

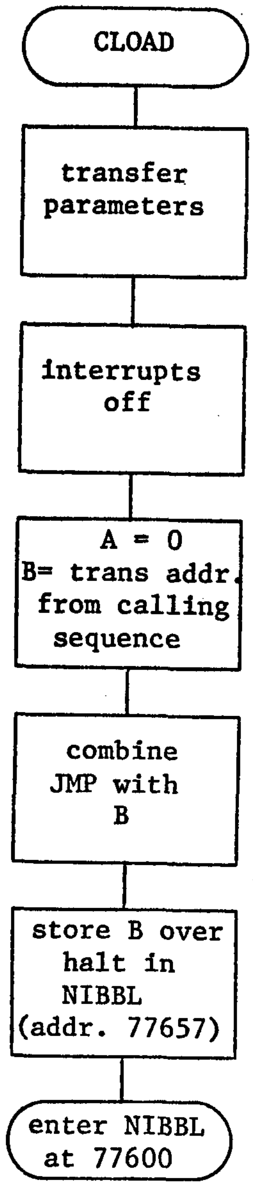
5. DESCRIPTION OF OUTPUT. None.

6. FILES USED. N/A

7. ERRORS AND RESTRICTIONS. An irrecoverable error will occur if a disc transfer error or a disc status error occurs.

8. COMPUTER OPERATOR INSTRUCTIONS. N/A

9. DESCRIPTION OF PROCESSING. CLOAD uses NIBBL, the Disc Basic Binary Loader which reads or writes an entire coreload. However, NIBBL is a manually operated program and normally halts upon completion rather than return to a calling program. Also, since all of core up to NIBBL will be overlaid by the new coreload, it would be impossible to return control to CLOAD. Therefore, CLOAD modifies NIBBL to return to an address specified in the CLOAD calling sequence. Processing is as follows: CLOAD turns off interrupts, combines the transfer address in the calling sequence with a JMP instruction, and stores this in NIBBL where the completion halt usually is. The cylinder number is picked up from the calling sequence, placed in the B register, and NIBBL is entered, with A=0. NIBBL will read the desired coreload and transfer control to the specified address.


CLOAD
transfer parameters
interrupts off
A = 0
B= trans addr. from calling sequence
combine JMP with B
store B over halt in NIBBL (addr. 77657)
enter NIBBL at 77600

```
0001                          ASMB,R,L
0003* THIS SUBROUTINE BRINGS IN THE 32K CORE LOAD WHICH BEGINS
0004* AT THE CYLINDER SPECIFIED AND STARTS EXECUTION AT ADDRESS 2.
0005* IT WORKS BY REMOVING THE HALT IN NIBBL AND THEN JUMPING TO IT.
0006* NIBBL MUST BE IN CORE FOR THIS TO WORK.  IT IS NORMALLY BROUGHT
0007* IN BY THE BOOT AT 77750.
0008*
0009* CALLING SEQUENCE:       JSB CLOAD
0010*                         DEF *+2
0011*                         DEF CYLNO
0012*                         DEF TRANS
0013*                            -OR-
0014*                    CALL CLOAD (CYLNO,TRANS)
0015*
0016  00000
0017                            NAM CLOAD
0018                            ENT CLOAD
0019                            EXT .ENTR
0019  00000 000000       CYLNO  NOP
0020  00001 000000       TRANS  NOP
0021  00002 000000       CLOAD  NOP
0022  00003 016001X             JSB .ENTR
0023  00004 000000R             DEF CYLNO
0024  00005 103100              CLF 0
0025  00006 002400              CLA
0026  00007 066014R             LDB JMP
0027  00010 146001R             ADB TRANS,I
0028  00011 176016R             STB HLTAD,I
0029  00012 166000R             LDB CYLNO,I
```

```
0030  00013 126015R        JMP NIBBL,I
0031  00014 024000  JMP    JMP 0
0032  00015 077600  NIBBL  OCT 77600
0033  00016 077657  HLTAD  OCT 77657
0034                       END
**  NO ERRORS *TOTAL **RTE ASMB 750420**



/ASMB: $END
:
```

DMA1

This assembly language program is used to transfer major frame ERAPS active data, under the control of Y7896. DMA is used in the data transfer from interface 23B.

FORTRAN calling sequence:

CALL DMA1 (IDAT, KK)

where IDAT = 72 word array

KK =

- 1 Turn off interrupts, turn on DMA. Return to main program.
- 2 Wait for DMA complete. Transfer 72 word major frame to IDAT. Return to main program.
- 3 Wait for DMA complete, turn on interrupts then return to main program

```
DMA1S  T=00004 IS ON CR00001 USING 00008 BLKS R=0084

ASMB,R,L
       HED DMA1
       NAM DMA1
       ENT DMA1
       EXT .ENTR
*
IDAT   NOP
KK     NOP
DMA1   NOP              ENTRY - FORTRAN CALLABLE
       JSB .ENTR
       DEF IDAT
       LDA KK,I
       CPA =D2
       JMP KK2
       CPA =D3
       JMP KK3
*
       CLF 0            KK=1, INTERRUPTS OFF
       LDA BUF1A        START INPUT
       JSB DMAI         INIT DMA
       STC SC,C
       STC 6,C          START DMA
       CLA,INA
       OTA SC           OUTPUT A 1 TO START
       JMP DMA1,I       RETURN
*
KK2    SFS 6            KK=2, SWAP BUFFERS AND CONTINUE
       JMP *-1
       CLC SC
       CLC 6
       LDA BUF2A        SWAP BUFFERS
       LDB BUF1A
```

```
       STA BUF1A
       STB BUF2A
       JSB DMAI
       STC SC,C
       STC 6,C        START DMA
       LDB =D-72      MOVE LAST SET OF DATA
       LDA BUF2A
       STA PTR1
       LDA IDAT
       STA PTR2
MVNXT  LDA PTR1,I
       STA PTR2,I
       ISZ PTR1
       ISZ PTR2
       ISZ B
       JMP MVNXT      MOVE NEXT
       JMP DMA1,I     RETURN
*
KK3    SFS 6          KK=3, WAIT FOR DMA, STOP
       JMP *-1
       CLC SC
       CLC 6
       CLA
       OTA SC
       STF 0          INTERRUPTS ON
       JMP DMA1,I     RETURN
*
DMAI   NOP            INIT DMA SUBROUTINE
       STA B
       LDA SLCT2
       IOR =B120000
       OTA 6
       CLC 2
       LDA B
       IOR =B100000
```

```
              OTA  2
              STC  2
              LDA  =D-72
              OTA  2
              JMP  DMAI,I
      *
      PTR1    NOP
      PTR2    NOP
      BUF1A   DEF  BUF1
      BUF2A   DEF  BUF2
      BUF1    BSS  72
      BUF2    BSS  72
      SLCT2   DEF  SC
      SC      EQU  23B
      A       EQU  0
      B       EQU  1
              END

:
```

```
D750S  T=00004 IS ON CR00001 USING 00002 BLKS R=0016

ASMB,R,L
        NAM D750
        ENT D750
D750    NOP
        ISZ D750
        LDA K1
        STA CNTR
DLY1    LSL 16
        LSL 16
        LSL 16
        ISZ CNTR
        JMP DLY1
        JMP D750,I
K1      DEC -29435
CNTR    NOP
        END

:
```

USER DATA ENTRY PROGRAM - INPT

1. FUNCTION. INPT provides a means to enter new analysis parameters or to use previously stored values in the analysis program Y1002.

2. CONSTRAINTS. All input variables must be integers.

3. CALLING SEQUENCE.

Call INPT (IVAR,IMIN,IMAX)

where: IVAR is parameter value

IMIN is minimum permissible value

IMAX is maximum permissible value

4. DESCRIPTION OF INPUT. Input is a typed value entered by operator on keyboard, or a carriage return.

5. DESCRIPTION OF OUTPUT. Printed output in parentheses is previously stored value. Upon carriage return or entry of new value, value is passed to calling program.

6. FILES USED. N/A

7. ERRORS AND RESTRICTIONS. Error and range checking is provided by the subroutine.

8. COMPUTER OPERATOR INSTRUCTIONS. N/A.

9. DESCRIPTION OF PROCESSING. The purpose of the subroutine INPT is to read in analysis constraints, if within the proper range of values of IMIN and IMAX, or to use a previously entered value, if a carriage return is hit.

At (1), the previously entered value is written on the CRT screen in parentheses.

At (2), the subroutine TYPI is used to fetch information entered from the keyboard. Two arguments are passed, the new value IN, and a parameter I1. I1 is equal to zero f a carriage return has been hit, indicating the user elected to use the previous value, and positive, if the user entered a new value IN.

At (3), the condition of a carraige return being hit is checked, and a return to the calling program is made if it is true.

At (4), the new value entered, IN, is checked for equivalence to 12345. If so, a transfer is made to the executive program in coreload 0.

At (5), the newly entered value, IN, is checked to be within the range of IMIN and IMAX. If out of range, an error message is written and the program branches to (2) to accept a new entry. If no errors are detected, the value IVAR is set to IN, at (6). The subroutine then returns to the calling program.

INPTS  T=00004 IS ON CR00001 USING 00004 BLKS R=0036

```
FTN4,L
      SUBROUTINE INPT(IVAR,IMIN,IMAX)
C
C     THIS ROUTINE IS USED TO BUFFER USER-ENTERED DATA
C     FOR USE IN THE OLAAS PROGRAM. DATA IS CHECKED FOR
C     ERRORS IN ENTRY AND IF NO DATA IS ENTERED, THE PREVIOUS
C     VALUE OF THE PARAMETER WILL BE USED.
C
      WRITE(2,100) IVAR
  100 FORMAT("(",I5,")")
    5 CALL TYPI(IN,I1)
C
C     RETURN IF CARRIAGE RETURN ONLY
C
      IF(I1 .EQ.0) RETURN
C
C     CHECK FOR RETURN TO EXECUTIVE (IN= 12345)
C
      IF(IN .EQ. 12345)CALL CLOAD(1,2)
C
C     CHECK FOR BAD DATA
C
      IF(I1 .LT.0 .OR. IN .GT. IMAX .OR. IN .LT. IMIN) GO TO 30
C
C     STORE NEW GOOD VALUE AS IVAR
C
      IVAR=IN
      RETURN
C
C     ERROR DETECTED, GET NEW INPUT
C
   30 WRITE(2,150)
```

```
INPTA  T=00004 IS ON CR00001 USING 00004 BLKS R=0036

FTN4,L
      SUBROUTINE INPT(IVAR,IMIN,IMAX)
C
C     THIS ROUTINE IS USED TO BUFFER USER-ENTERED DATA
C     FOR USE IN THE OLAAS PROGRAM. DATA IS CHECKED FOR
C     ERRORS IN ENTRY AND IF NO DATA IS ENTERED, THE PREVIOUS
C     VALUE OF THE PARAMETER WILL BE USED.
C
      WRITE(2,100) IVAR
  100 FORMAT("(",I5,")")
    5 CALL TYPI(IN,I1)
C
C     RETURN IF CARRIAGE RETURN ONLY
C
      IF(I1 .EQ.0) RETURN
C     CHECK FOR BAD DATA
C
      IF(I1 .LT.0 .OR. IN .GT. IMAX .OR. IN .LT. IMIN) GO TO 30
C
C     STORE NEW GOOD VALUE AS IVAR
C
      IVAR=IN
      RETURN
C
C     ERROR DETECTED, GET NEW INPUT
C
   30 WRITE(2,150)
  150 FORMAT("ERROR, REENTER VALUE")
      GO TO 5
      END
      END$
```

DATA COLLECTION PROGRAM OPERATOR INPUT ROUTINE - INPUT

1. FUNCTION. INPUT provides conversational input and error checking of operator parameters for the data collection program.

2. CONSTRAINTS. All input variables must be integers.

3. CALLING SEQUENCE. Called by: Y0001

```
JSB INPUT
DEF *+7
DEF N
DEF K
DEF L
DEF M
DEF D
DEF S
```

where N is pulse time (1-240)
K is delay (5-60)
L is number of beams (1-16)
M is number of pings (1-100)
D is desired direction (0-360)
S is source (0-1)

4. DESCRIPTION OF INPUT. Input is typed values in the order described in the calling sequence.

5. DESCRIPTION OF OUTPUT. Output is the variables described in the calling sequence, in integer format. Printed output is the operator prompts.

6. FILES USED. N/A

7. ERRORS AND RESTRICTIONS. None.

8. COMPUTER OPERATOR INSTRUCTIONS. See general system operating instructions.

9. DESCRIPTION OF PROCESSING. INPUT requests input of each variable by a message on the CRT. TYPI is called to input and convert the value from the keyboard. If TYPI returns an error flag, or no data is entered, or the value is out of range, the prompt is given again. Each variable is input in sequence by the same method.

MPINP  T=00004 IS ON CR00001 USING 00029 BLKS R=0160

```
FTN4
      SUBROUTINE INPUT (N,K,L,M,D,S)

C  *************************************************************
C  *                                                           *
C  * TITLE:  INPUT QUESTION/ANSWER SUBROUTINE FOR THE          *
C  *         OLAAS DATA COLLECTION SUBROUTINE                  *
C  *                                                           *
C  * MODULE:  INPUT      SOURCE:  MPINP      VERSION:  X01     *
C  *                     OBJECT:  $MPINP                       *
C  *                                                           *
C  * AUTHOR:  R. TASHJIAN        DATE:  1976                   *
C  *                                                           *
C  * MODIFICATIONS:                                            *
C  *   1:  CHANGE THE QUESTION FORMATS.                        *
C  *         12/14/77     W. BOWEN                             *
C  *                                                           *
C  * MODULE FUNCTION:                                          *
C  *    THE INPUT QUESTION/ANSWER (INPUT) SUBROUTINE IS USED   *
C  * TO PLACE THE DATA COLLECTION PARAMETER QUESTIONS ON THE   *
C  * SCREEN.   THE PREVIOUS PARAMETER VALUES ARE GIVEN.   THE  *
C  * USER ENTERS NEW PARAMETERS OR USES THE PREVIOUS           *
C  * PARAMETER VALUES.    ENTRY DATA IS CHECKED FOR RANGE      *
C  * ERRORS.                                                   *
C  *                                                           *
C  * CALLING SEQUENCE:     CALLED BY - Y0001                   *
C  *                                                           *
C  *     CALL INPUT (N,K,L,M,D,S)                              *
C  *                                                           *
C  * INPUT/OUTPUT PARAMETERS:  N = PULSE TIME (AFTER K)        *
C  *                           K = DELAY BEFORE RECORD         *
C  *                           L = NUMBER BEAMS TO RECORD      *
```

```
C  *                                        M = TOTAL NUMBER OF PINGS       *
C  *                                        D = DESIRED DIRECTION OF        *
C  *                                            CENTER BEAM                 *
C  *                                        S = SOURCE (1 = ACTIVE ON-LINE  *
C  *                                                    0 = TAPE RECORDER   *
C  *                                                                        *
C  ************************************************************************

C  ************************************************************************
C  *  DEFINE ARRAYS AND VARIABLES                                         *
C  ************************************************************************

      DIMENSION IPBUF(32)

      INTEGER D,S

C  ************************************************************************
C  *  INITIALIZE SUBROUTINE                                               *
C  ************************************************************************

C > > > > CLEAR THE PRINT BUFFER AREA
      IBUFL =64
      DO 5 I=1,32
      IPBUF(I) = 020040B
    5 CONTINUE
      WRITE (2,8)
    8 FORMAT (/5X,"** RAW DATA RECORDING QUESTIONS  **")

C > > > > DETERMINE QUESTION ENTRY POINT
      IQUE = 0
  100 IQUE = IQUE + 1
      GO TO (10,20,30,40,50,60) IQUE

C  ************************************************************************
C  *  DATA COLLECTION PARAMETER QUESTIONS                                 *
```

```
C   ************************************************************
 
C > > > > START RANGE
   10 IMIN = 5
      IMAX = 35
      MSTRT = 8
      CALL CODE
      WRITE (IPBUF,15) MSTRT
   15 FORMAT ("START RANGE",8X,"IN KILOYARDS  (5 TO 35)"
     +13X,"(",I5,")",2X)
        GO TO 500
 
C > > > > STOP RANGE
   20 IMIN = MSTRT
      IMAX = 40
      MEND = 24
      CALL CODE
      WRITE (IPBUF,25) MEND
   25 FORMAT ("STOP RANGE",9X,"IN KILOYARDS  (MAXIMUM 40)",
     +10X,"(",I5,")",2X)
        GO TO 500
 
C > > > > CONVERT TO CORRESPONDING TIME IN SECONDS
 
   30 X = MSTRT * 1.2
      K = X + 0.5
      X = (MEND -MSTRT) * 1.2 + 3.0
      N = X + 0.5
 
C > > > > NUMBER OF BEAMS TO RECORD
      IMIN = 1
      IMAX = 16
      L = 16
      CALL CODE
      WRITE (IPBUF,35) L
```

```
   35 FORMAT ("BEAMS",14X,"NUMBER OF BEAMS TO RECORD",
     +11X,"(",I5,")",2X)
        GO TO 500

C > > > > NUMBER OF PINGS TO RECORD
   40 IMIN = 1
      IMAX = 100
      M = 1
      CALL CODE
      WRITE (IPBUF,45) M
   45 FORMAT ("NUMBER PINGS",7X,"NUMBER OF PINGS TO RECORD",
     "11X,"(",I5,")",2X)
        GO TO 500

C >>>>>>> SKIP NEXT QUESTION IF ALL BEAMS ARE RECORDED >>>>>>>
   50 IF (L .EQ. 16) GO TO 100

C > > > > DESIRED DIRECTION OF CENTER BEAM
      IMIN = 0
      IMAX = 359
      D = 0
      CALL CODE
      WRITE (IPBUF,55) D
   55 FORMAT ("CENTER BEAM",8X,"DESIRED CENTER BEAM DIRECTION",
     +7X,"(",I5,")",2X)
        GO TO 500

C > > > > DATA SOURCE
   60 IMIN = 0
      IMAX = 1
      S = 1
      CALL CODE
      WRITE (IPBUF,65) S
   65 FORMAT ("DATA SOURCE",8X,"1=ACTIVE ON-LINE / 0=",
     +"TAPE RECORDER",2X,"(",I5,")",2X)
```

```
      GO TO 500

C ************************************************************
C *  CALL PRINT LINE (PNTL) SUBROUTINE TO PRINT QUESTIONS     *
C ************************************************************

  500 CALL PNTL (IPBUF,IBUFL)

C ************************************************************
C *  CALL TO INPUT PARAMETERS FROM THE KEYBOARD               *
C ************************************************************

      CALL TYPI (IN,I1)

C > > > > RETURN IF CARRIAGE RETURN ONLY
      IF (I1 .EQ. 0) GO TO 510

C > > > > CHECK FOR RETURN TO EXECUTIVE  (IN = 12345)
      IF (IN .EQ. 12345) CALL CLOAD (1,2)

C > > > > CHECK IF DATA IS OUT OF RANGE  (IMIN TO IMAX)
      IF (I1 .LT. 0 .OR. IN .GT. IMAX .OR. IN .LT. IMIN) GO TO 600
C ************************************************************
C *  STORE GOOD VALUE IN PARAMETER VARIABLES                  *
C ************************************************************

      IF (IQUE .EQ. 1) MSTRT = IN
      IF (IQUE .EQ. 2) MEND = IN
      IF (IQUE .EQ. 3) L = IN
      IF (IQUE .EQ. 4) M = IN
      IF (IQUE .EQ. 5) D = IN
      IF (IQUE .EQ. 6) S = IN
  510 IF (IQUE .NE. 6) GO TO 100
```

```
C > > > > RETURN TO CALLER
      RETURN

C  **********************************************************************
C  *  ERROR DETECTED, GET NEW INPUT                                     *
C  **********************************************************************

  600 WRITE (2,605)
  605 FORMAT ("********** PARAMETER ERROR **********"
     +"      RE-ENTER VALUE")
      GO TO 500

C >>>>>>>>>>>>>>>>> END OF SUBROUTINE INPUT >>>>>>>>>>>>>>>>>
      END
      END$

:
```

```
ISHFTS T=00003 IS ON CR00001 USING 00003 BLKS R=0000

ASMB,R,L
        HED
        HED FUNCTION ISHFT(I,N)
        NAM ISHFT
        ENT ISHFT
        EXT .ENTR
*
* THIS FORTRAN FUNCTION SHIFTS I RIGHT N BITS WITH SIGN EXTEND.
*
I       NOP
N       NOP
ISHFT   NOP
        JSB .ENTR
        DEF I
        LDA N,I
        AND =B17
        IOR =B101020 MERGE INTO ASR INSTRUCTION
        STA ASR
        LDB I,I      GET IT
ASR     ASR 16       SHIFT IT
        LDA B        LEAVE IT
        JMP ISHFT,I  RETURN
A       EQU 0
B       EQU 1
        END

:
```

```
PAGE 0002 #01  TIME BASE GENERATOR INTERRUPT LINKAGE

0001                   ASMB,R,L
0003  00000                  NAM LKTBG    R.E.T. 8-4-76
0004                         ENT LKTBG
0005                         EXT TBGI
0006  00000 016001X LKTBG JSB TBGI
0007                         END
**  NO ERRORS *TOTAL **RTE ASMB 750420**
```

```
0001                    ASMB,R,L
0003  00000                   NAM LINK6    R.E.T. 8-4-76
0004                          ENT LINK6
0005                          EXT BFIT
0006* THIS PRIVIDES A BASE PAGE LINK TO THE BEAM FORMER (DMA)
0007* COMPLETION PROCESSOR.
0008  00000 016001X LINK6 JSB BFIT
0009                          END
**  NO ERRORS *TOTAL **RTE ASMB 750420**
```

RETRIEVE DATA SUBROUTINE - RDATA

1. FUNCTION. RDATA retreives data stored by Y0001 from the disc as specified by arguments in the call. A separate entry point, HEADR, retrieves the header data for a specified ping.

2. CONSTRAINTS. Although validity checks are made on ping and beam number and number of words, no check is made on the index specification, nor is any check made on whether the data was actually recorded.

3. CALLING SEQUENCE.

```
    CALL RDATA (IRPING,IBEAM,INDXP,NWDS,ISTAT)
```

where IPING is the ping number (1-?)

PBEAM is the beam number (1-16)

INDXP is the index (0-?)

NWDS is the number of words to be read (1-4096)

ISTAT is the location for the status word

INDXP specifies the first word to be read. A change in INDXP of 1 corresponds to 128 words. Therefore INDXP=5 indicates that the reading is to start with word 512. ISTAT is set to zero if data is returned, non-zero if no data is returned. Data is stored in Data Block 0 (zero).

```
    CALL HEADR (IPING,IDATA)
```

where IPING is the ping number

IDATA is a 16 word array where the header data is to be stored.

4. DESCRIPTION OF INPUT. Input is the information written on disc by the data collection program, Y0001. See Figure 6 in the Y0001 description for format.

5. DESCRIPTION OF OUTPUT. Output is the requested number of words of data for the specified ping. All beam data is placed in Data Block 0 (zero). This data is a series of 2's complement, fixed-point, single-word variables of like scaling. Output from subroutine HEADR is as follows. All data items are integer format, with the exception of time, which is BCD.

| | | | |
|---|---|---|---|
| Word | 1 | TIME | time code, hours, BCD |
| | 2 | | time code, minutes and seconds, BCD |
| | 3 | ANGL1 | heading of beam 1, degrees |

| | | | |
|---|---|---|---|
| Word | 4 | | not used |
| | 5 | FSTB# | number of first beam recorded |
| | 6 | FBANG | heading of first beam recorded, degrees |
| | 7 | PING | ping number |
| | 8 | #BMS | number of beams recorded |
| | 9 | S | source (1=on line, 0=recorder) |
| | 10 | TOTPG | total number of pings (also known as M) |
| | 11 | K | delay before record, seconds |
| | 12 | N | length of record, seconds |
| | 13 | | reserved |
| | 14-16 | | not used |

6. FILES USED. N/A

7. ERRORS AND RESTRICTIONS. None.

8. COMPUTER OPERATOR INSTRUCTIONS. N/A

9. DESCRIPTION OF PROCESSING. RDATA first transfers parameters from the calling sequence and turns interrupts off. It sets the status parameter bad so that if no data is read, it will be set. If HEADR has never been called, certain parameters essential for locating the beam data will not have been calculated; INIT=0 indicates this. If INIT is zero, HEADR is called to read the ping 1 header into the local buffer and calculate the number of sectors per block (SPB) and per ping (SPP). INIT is set to 1.

At (2) the caller's ping number is checked for legality against the maximum ping number given in the header record. It should be noted that the header record being referenced at this point may be any header record, but maximum ping number is the same for all. Next, the requested ping number is checked against the number of the ping in the buffer. If different, HEADR is called again to read the requested ping header into the local buffer. It is always necessary to have the proper ping's header in core before reading beam data, as the number of the first beam (FSTB#) is needed in order to determine the position of the requested beam. The ping number check before reading the header avoids unnecessary disc accesses when many requests are made for the same ping.

At (3) the requested beam number is checked for being between FSTB# and FSTB#+#BMS-1. Some special processing is required due to the "wraparound" nature of the beam numbers; i.e., beam 1 is adjacent to beam 16.

Beginning at (4) the starting sector for the requested ping, beam, and index is computed. This also takes into account that there is a header record (1 sector) at the beginning of each ping. This point of the subroutine makes use of a special sector number, SSECT, which does not reset to zero at the beginning of each track. Therefore for track 0 sector 0, SSECT would be 0, and for track 1 sector 0, SSECT would be 48. SSECT is also relative and does not account for the actual first track on the disc. SSECT will later be converted to actual track and sector by RDISC. The computation of SSECT also includes computation of OFST, the offset within a block of 4 sectors of the same beam. This is necessary because unless the index specifies starting at a certain location the first disc read will not include a full 4 sectors. The counter CNTR is initialized to DTADO, which is within the Fourier system.

Finally at (5) the number of words to the end of the first block (RWDS) is calculated; this will be the number of words in the read request. If RWDS is greater than the number of words remaining to be read (-CNTR) the smaller number replaces RWDS. RDISC is called to read RWDS words directly into DTADO at the starting location given by PNTR. PNTR and CNTR are updated by the number of words read, RWDS. If CNTR is then still not zero, the special sector SSECT is updated to the beginning of the next block of the applicable beam. Since the next read will begin at the beginning of the 4 sector block, OFST is cleared. Control is then transferred back to (5) to read the next block. When CNTR finally reaches zero, the status indicator is set to zero (good), interrupts are turned on and control is returned to the caller.

The HEADR subroutine is called both by the user program to read the header into a user array, and by RDATA to read the header into the local buffer. As a result, the interrupt system status (on for user, off for RDATA) is unknown at entry. Therefore RDATA first determines this condition and saves it in INTST, then turns the interrupt system off. The data location from the calling sequence is set up in the RDISC calling sequence.

At (6) INIT is checked. If zero, there has been no header read into the local buffer, which is necessary to locate other data on the disc. In this case, SSECT is set to zero to specify the beginning of ping 1, which is the header, RDISC is called to read 16 words at SSECT into the local header buffer, and INIT is set to 1 to eliminate the reading of ping 1 header in the future.

At (7) the number of sectors per block

SPB = #BMS * 4

and number of sectors per ping

SPP = SPB*BLOCK + 1

are computed. The 1 in the second equation accounts for the header at the beginning of each ping. The starting special sector (SSECT) for the requested ping, which is the header location for that ping, is computed and RDISC is called to read 16 words, the header, from that location into the user's buffer. Finally, HEADR restores the interrupt system condition and returns.

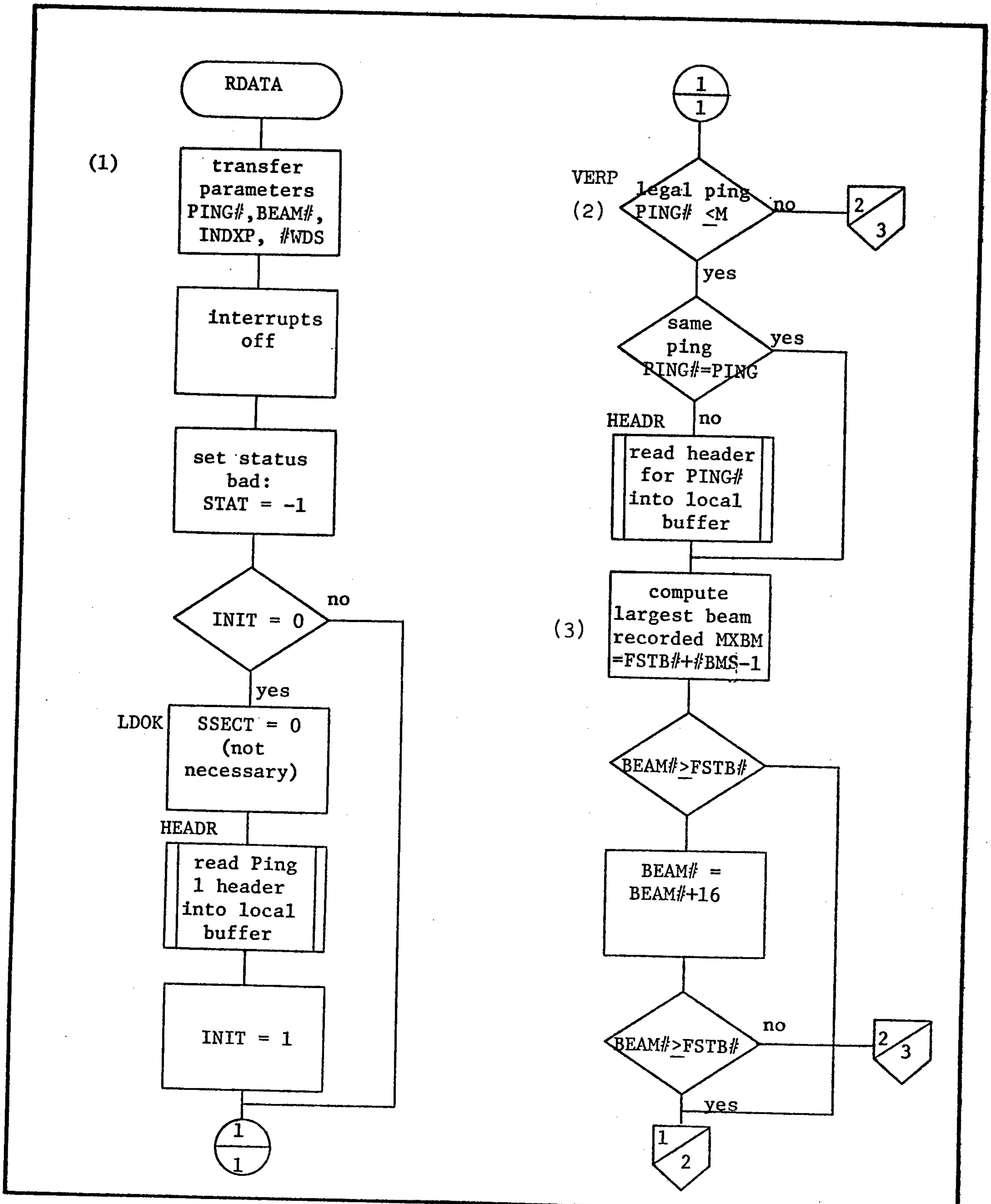
Subroutine RDISC gets its information on disc location from the special sector indicator, SSECT, as explained in the RDATA description. RDATA first converts this to a relative track and sector by dividing by 48. The quotient is stored in STRAK and the remainder is stored in SECTO. STRAK is relative to the first allocated track on the disc. SECT has a range of 0-47, with the numbers 24-47 representing sectors 0-23 on the upper surface, as described for WDISC (a part of Y0001). The actual track (TRAK) is then computed based on STRAK and the Track Allocation Table entries: FTLD, LTLD, FTUD, and LTUD. If the final track number is greater than LTUD or there is no disc space allocated, an error exit is taken. Along with the track calculations, the head base (HEADB) is set up according to the disc in use. HEADB corresponds to the lower head for the disc in use. DISCR will use a head one greater if SECT specifies the upper surface.

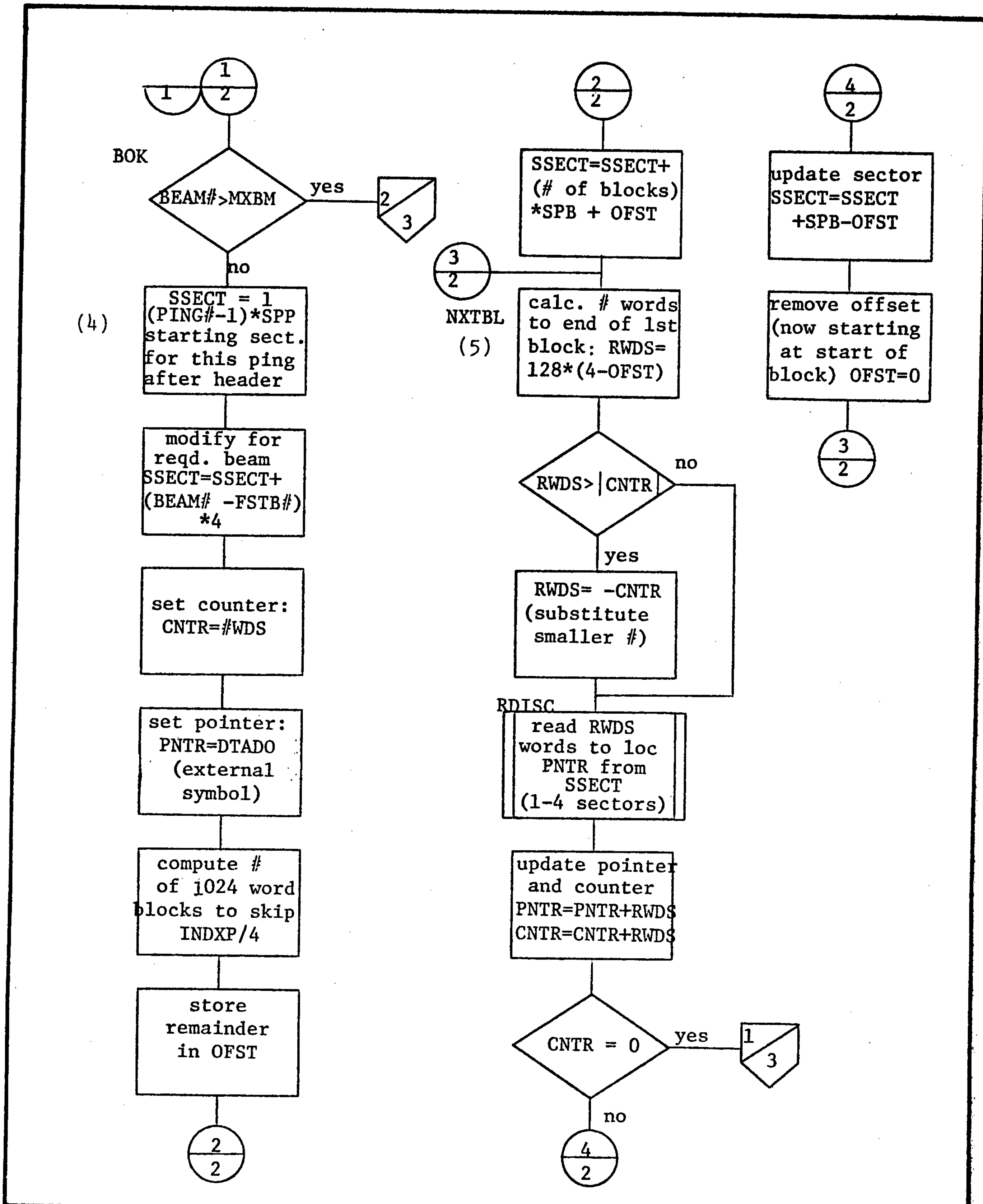
At (8) the number of words remaining on the track, starting at SECT, is calculated. This is significant because a disc access can not cross a track boundary; i.e., head motion commands must be explicitly given by the program. This is stored in AA. If RCTR is smaller, RCTR is stored in AA, and the access will not be to end of track. DISCR is called to perform the disc read, which will be AA words from TRAK and SECT on the disc specified by HEADB, into the location stored at RPTR.

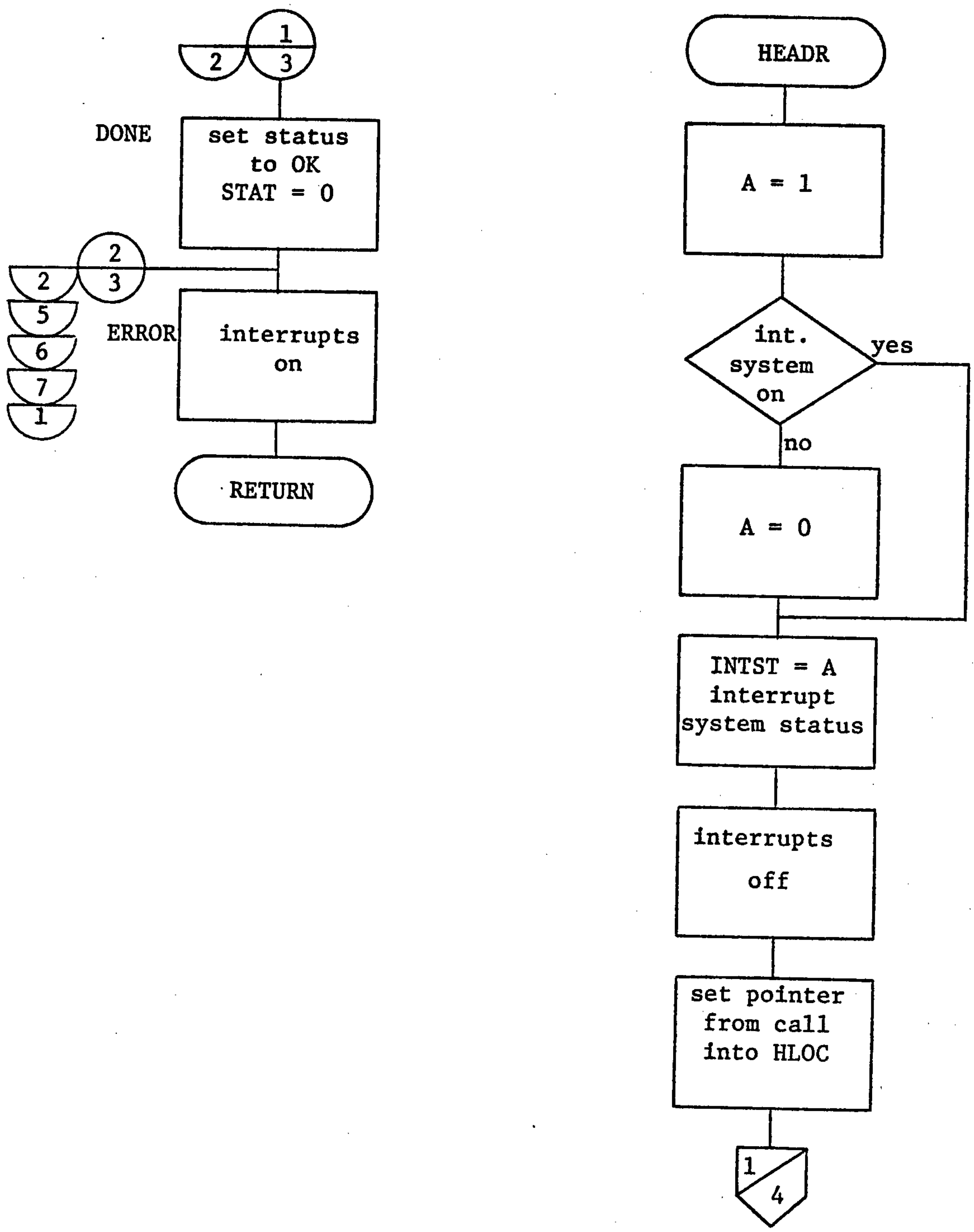
At (9) SECT is updated. If the access just completed was to end of track, SECT will now be 48 and RDISC will reset SECT to zero and increment TRAK. If TRAK is greater than MXTRK, TRAK and MXTRK are set to the upper disc track parameters, FTUD and LTUD, respectively. If HEADB is zero, the disc in use was already the upper disc, so an error exit is taken. Otherwise, HEADB is set to zero.

At (10) the pointer and word counter, RPTR and RCTR are updated by the word counter, RPTR and RCTR are updated by the word count in the last access, AA. If RCTR is then zero, all disc access requested by RDISC has been completed, and it returns to the calling program. Otherwise, control branches back to (8) for another read.

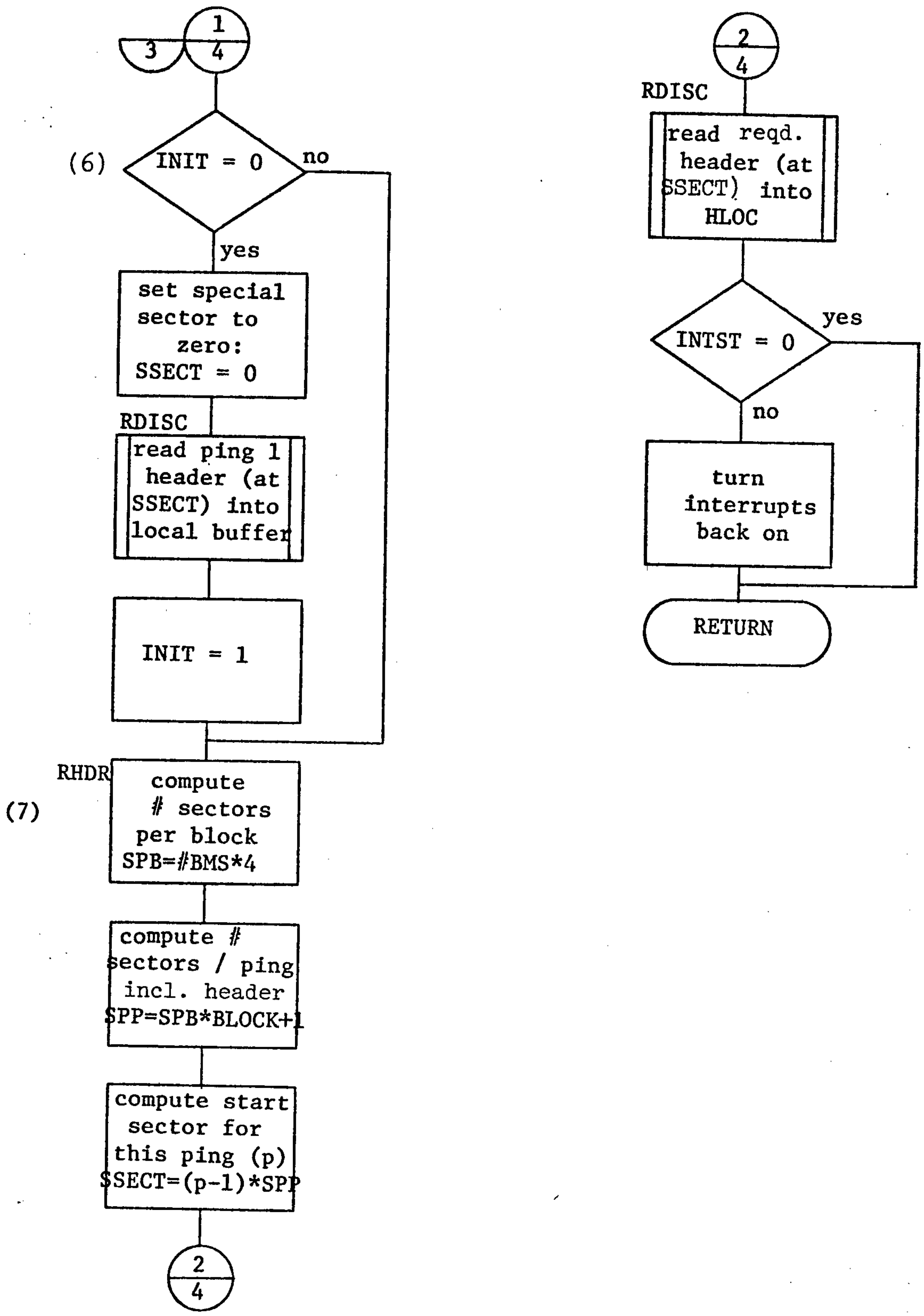
DISCR, called by RDISC, first converts SECT (range 0-47) and HEADB to HDSEC, the control word for head and sector which uses sector in the range 0-23. This is shifted to the bit positions required by the disc controller. DMA is set up to read AA words starting at the location in RPTR. The remainder of the processing is as specified in the disc interface manual. It should be noted that the order of commands for disc read is different from disc write, therefore there are some significant differences from WRITD. If status following the read is bad, an error path is taken.

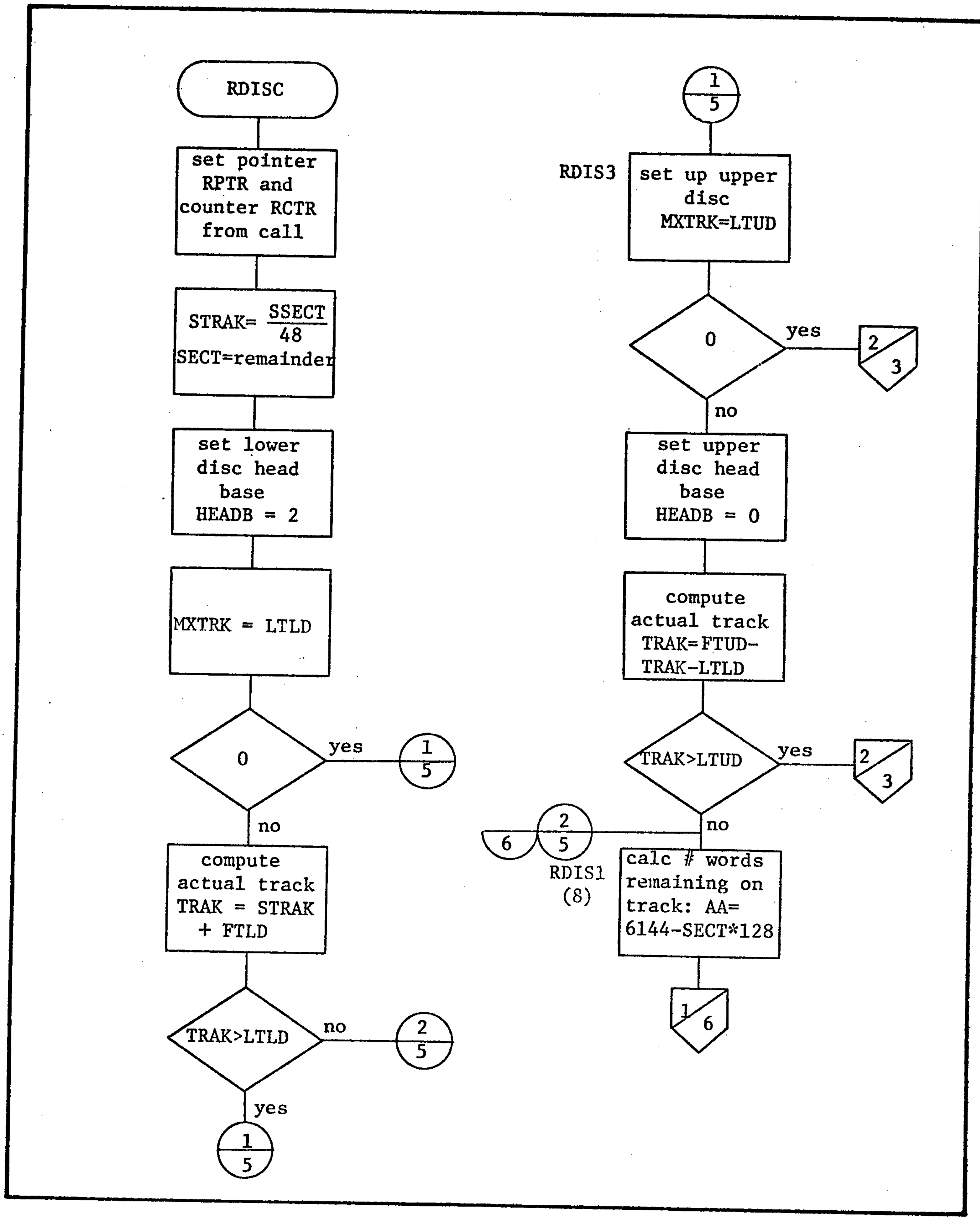

RDATA
(1)
transfer parameters PING#,BEAM#, INDXP, #WDS
interrupts off
set status bad: STAT = -1
INIT = 0
no
yes
LDOK
SSECT = 0 (not necessary)
HEADR
read Ping 1 header into local buffer
INIT = 1
1
1
1
1
VERP
(2)
legal ping PING# ≤M
no
2
3
yes
same ping PING#=PING
yes
HEADR
no
read header for PING# into local buffer
(3)
compute largest beam recorded MXBM =FSTB#+#BMS-1
BEAM#>FSTB#
BEAM# = BEAM#+16
BEAM#>FSTB#
no
2
3
yes
1
2

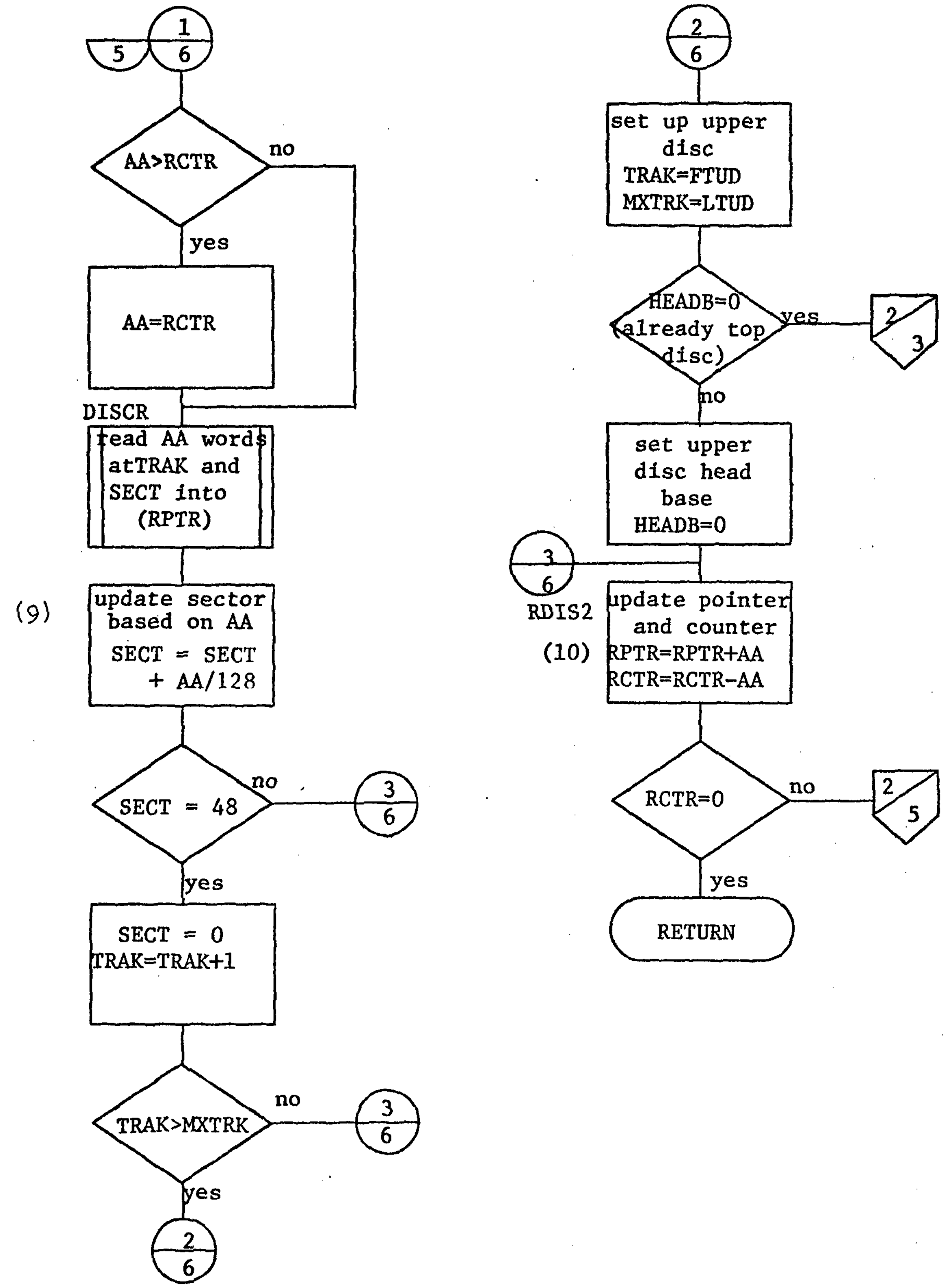

1
1
2
BOK
BEAM#>MXBM
yes
2
3
no
SSECT = 1
(PING#-1)*SPP
starting sect.
for this ping
after header
(4)
modify for
reqd. beam
SSECT=SSECT+
(BEAM# -FSTB#)
*4
set counter:
CNTR=#WDS
set pointer:
PNTR=DTADO
(external
symbol)
compute #
of 1024 word
blocks to skip
INDXP/4
store
remainder
in OFST
2
2
2
2
SSECT=SSECT+
(# of blocks)
*SPB + OFST
3
2
NXTBL
(5)
calc. # words
to end of 1st
block: RWDS=
128*(4-OFST)
RWDS>|CNTR
no
yes
RWDS= -CNTR
(substitute
smaller #)
RDISC
read RWDS
words to loc
PNTR from
SSECT
(1-4 sectors)
update pointer
and counter
PNTR=PNTR+RWDS
CNTR=CNTR+RWDS
CNTR = 0
yes
1
3
no
4
2
4
2
update sector
SSECT=SSECT
+SPB-OFST
remove offset
(now starting
at start of
block) OFST=0
3
2

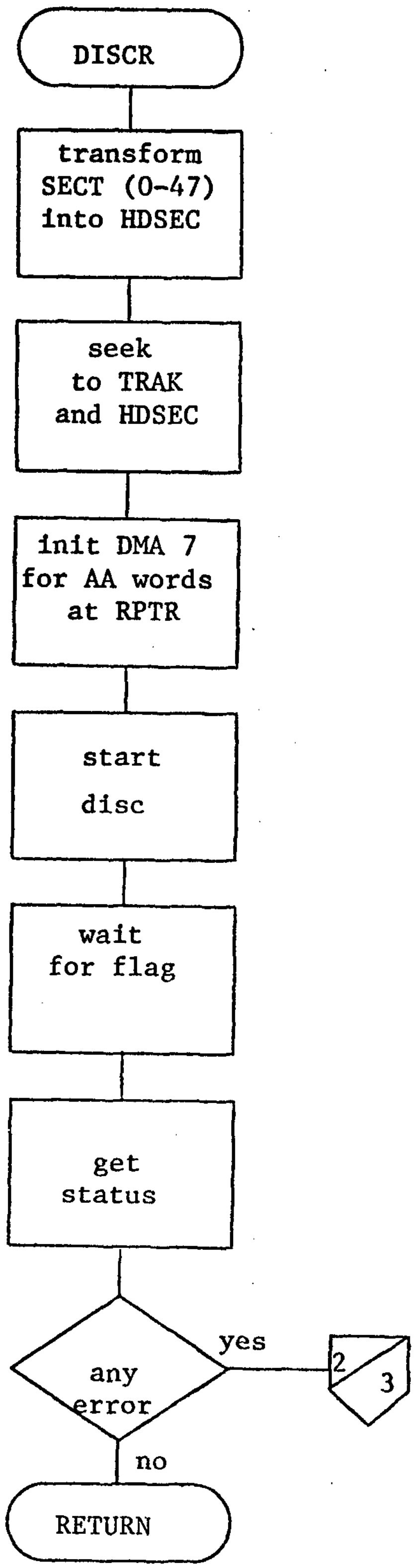


1
2
3
DONE
set status to OK STAT = 0
2
3
2
5
6
7
1
ERROR
interrupts on
RETURN
HEADR
A = 1
int. system on
yes
no
A = 0
INTST = A interrupt system status
interrupts off
set pointer from call into HLOC
1
4


1
4
3
(6)
INIT = 0
no
yes
set special sector to zero: SSECT = 0
RDISC
read ping 1 header (at SSECT) into local buffer
INIT = 1
RHDR
(7)
compute # sectors per block SPB=#BMS*4
compute # sectors / ping incl. header SPP=SPB*BLOCK+1
compute start sector for this ping (p) SSECT=(p-1)*SPP
2
4
2
4
RDISC
read reqd. header (at SSECT) into HLOC
INTST = 0
yes
no
turn interrupts back on
RETURN


RDISC
set pointer RPTR and counter RCTR from call
STRAK= SSECT/48
SECT=remainder
set lower disc head base HEADB = 2
MXTRK = LTLD
0
yes
no
compute actual track TRAK = STRAK + FTLD
TRAK>LTLD
no
yes
RDIS3
set up upper disc MXTRK=LTUD
0
yes
no
set upper disc head base HEADB = 0
compute actual track TRAK=FTUD-TRAK-LTLD
TRAK>LTUD
yes
no
RDIS1
(8)
calc # words remaining on track: AA= 6144-SECT*128


1
6
5
AA>RCTR
no
yes
AA=RCTR
DISCR
read AA words atTRAK and SECT into (RPTR)
(9)
update sector based on AA
SECT = SECT + AA/128
SECT = 48
no
3
6
yes
SECT = 0
TRAK=TRAK+1
TRAK>MXTRK
no
3
6
yes
2
6
2
6
set up upper disc
TRAK=FTUD
MXTRK=LTUD
HEADB=0 (already top disc)
yes
2
3
no
set upper disc head base
HEADB=0
3
6
RDIS2
(10)
update pointer and counter
RPTR=RPTR+AA
RCTR=RCTR-AA
RCTR=0
no
2
5
yes
RETURN


DISCR
transform SECT (0-47) into HDSEC
seek to TRAK and HDSEC
init DMA 7 for AA words at RPTR
start disc
wait for flag
get status
any error
yes
2
3
no
RETURN

```
PAGE 0002 #01   RDATA SUBROUTINE

0001                      ASMB,R,L
0003  00000                     NAM RDATA
0004                            ENT RDATA,HEADR
0005                            EXT .ENTR,DTAD0
0006                            EXT FTLD,LTLD,FTUD,LTUD
0007*
0008* THIS SUBROUTINE RETRIEVES DATA FROM DISC AS SPECIFIED
0009* BY ARGUMENTS IN CALL.
0010*
0011  00000 000000  PING# NOP           PING (1-7)
0012  00001 000000  BEAM# NOP           BEAM (1-16)
0013  00002 000000  INDXP NOP           INDEX (0-7, 1 IS 128 WORDS)
0014  00003 000000  #WDS  NOP           COUNT (1024 OR 2048)
0015  00004 000000  STAT  NOP           STATUS (0 IF DATA RETURNED)
0016  00005 000000  RDATA NOP
0017  00006 016001X       JSB .ENTR
0018  00007 000000R       DEF PING#
0019  00010 162000R       LDA PING#,I
0020  00011 072000R       STA PING#
0021  00012 162001R       LDA BEAM#,I
0022  00013 072001R       STA BEAM#
0023  00014 162002R       LDA INDXP,I
0024  00015 072002R       STA INDXP
0025  00016 162003R       LDA #WDS,I
```

```
00017 072003R          STA #WDS
*
00020 103100           CLF 0         INTERRUPTS OFF
00021 003400           CCA
00022 172004R          STA STAT,I    SET STATUS BAD
*
00023 062201R          LDA INIT
00024 002002           SZA           POINTERS INITIALIZED? (=1)
00025 026036R          JMP VERP      YES, SKIP READ OF HEADER DATA
*
00026 002400    LDOK   CLA           READ HEADER DATA
00027 072203R          STA SSECT
00030 016234R          JSB HEADR
00031 000034R          DEF *+3
00032 000200R          DEF D1
00033 000212R          DEF LBUF+1
00034 002404           CLA,INA       SET INIT=1 TO SKIP HEADER READ
00035 072201R          STA INIT        NEXT TIME
*
00036 062000R   VERP   LDA PING#
00037 003004           CMA,INA
00040 042223R          ADA M
00041 002020           SSA           PING# .LE. NUMBER OF PINGS?
00042 026176R          JMP ERROR     NO, ERROR
*
00043 062000R          LDA PING#
00044 052220R          CPA PING
00045 026052R          JMP VERB      PING SAME AS LAST, JUMP
00046 016234R          JSB HEADR
00047 000052R          DEF *+3
00050 000000R          DEF PING#
00051 000212R          DEF LBUF+1
*
00052 062216R   VERB   LDA FSTB#     VERIFY BEAM # WITHIN LIMITS
00053 042221R          ADA #BMS
```

```
0061  00054 042553R        ADA =D-1
0062  00055 072205R        STA MXBM     LARGEST BEAM # (SOMETIMES +16)
0063  00056 062216R        LDA FSTB#
0064  00057 003004         CMA,INA
0065  00060 042001R        ADA BEAM#
0066  00061 002021         SSA,RSS
0067  00062 026073R        JMP BOK      BEAM# .GE. FSTB#
0068  00063 062554R        LDA =D16     BEAM# .LT. FSTB#, ADD 16
0069  00064 042001R        ADA BEAM#
0070  00065 072001R        STA BEAM#
0071  00066 062216R        LDA FSTB#        AND TRY AGAIN
0072  00067 003004         CMA,INA
0073  00070 042001R        ADA BEAM#
0074  00071 002020         SSA          BEAM# .LT. FSTB# ?
0075  00072 026176R        JMP ERROR    YES, ERROR
0076*
0077  00073 062001R BOK    LDA BEAM#    CHECK FOR BEAM TOO LARGE
0078  00074 003004         CMA,INA
0079  00075 042205R        ADA MXBM
0080  00076 002020         SSA          BEAM# .GT. MXBM ?
0081  00077 026176R        JMP ERROR    YES, ERROR
0082*
0083  00100 062000R        LDA PING#
0084  00101 042553R        ADA =D-1
0085  00102 100200         MPY SPP      STARTING SECTOR FOR THIS PING
      00103 000207R
0086  00104 002004         INA          SKIP OVER HEADER DATA
0087  00105 072203R        STA SSECT
0088*
0089  00106 062216R        LDA FSTB#
0090  00107 003004         CMA,INA
0091  00110 042001R        ADA BEAM#
0092  00111 100200         MPY =D4
      00112 000555R
0093  00113 042203R        ADA SSECT    STARTING SECTOR FOR DESIRED BEAM
```

```
0094   00114 072203R          STA SSECT
0095*
0096   00115 062003R          LDA #WDS     SET UP POINTERS AND COUNTER
0097   00116 003004           CMA,INA
0098   00117 072204R          STA CNTR
0099   00120 062002X          LDA DTAD0
0100   00121 072152R          STA PNTR
0101*
0102   00122 062002R          LDA INDXP    COMPUTE HOW MANY SETS OF 1024
0103   00123 006400           CLB            WORDS TO SKIP OVER (# BLOCKS)
0104   00124 100400           DIV =D4
       00125 000555R
0105   00126 076210R          STB OFST     REMAINDER POINTS TO MID-BLOCK
0106   00127 100200           MPY SPB
       00130 000206R
0107   00131 042210R          ADA OFST
0108   00132 042203R          ADA SSECT
0109   00133 072203R          STA SSECT    STARTING SECTOR FOR DESIRED INDEX
0110*
0111   00134 062210R NXTBL    LDA OFST     READING OF ONE BEAM OF ONE BLOCK
0112   00135 003004           CMA,INA        STARTS HERE
0113   00136 042555R          ADA =D4      COMPUTE NO. CONTIGUOUS WORDS
0114   00137 100200           MPY =D128    (4-OFST) X 128
       00140 000556R
0115   00141 072153R          STA RWDS     # WORDS TO END OF THIS BLOCK
0116   00142 042204R          ADA CNTR
0117   00143 003004           CMA,INA
0118   00144 066204R          LDB CNTR     GET ABS OF CNTR
0119   00145 007004           CMB,INB
0120   00146 002020           SSA          RWDS .GT. ABS CNTR ?
0121   00147 076153R          STB RWDS     YES, MAKE RWDS SMALLER
0122   00150 016310R          JSB RDISC    READ RWDS WORDS AT SSECT TO PNTR
0123   00151 000154R          DEF *+3
0124   00152 000000  PNTR     NOP          LOCATION WITHIN DTAD0
0125   00153 000000  RWDS     NOP          # WORDS TO READ (UP TO 4 SECTORS)
```

```
00154 062153R        LDA RWDS     UPDATE POINTER AND COUNTER
00155 042152R        ADA PNTR       BY NUMBER OF WORDS READ
00156 072152R        STA PNTR
00157 062153R        LDA RWDS
00160 042204R        ADA CNTR
00161 072204R        STA CNTR
00162 002003         SZA,RSS      COUNTER=0?
00163 026174R        JMP DONE     YES, DONE
00164 062210R        LDA OFST     NO, REMOVE OFFSET FROM SSECT AND
00165 003004         CMA,INA        UPDATE TO NEXT BLOCK
00166 042203R        ADA SSECT
00167 042206R        ADA SPB
00170 072203R        STA SSECT
00171 002400         CLA          CLEAR OFFSET (NOW STARTING AT
00172 072210R        STA OFST       START OF BLOCK)
00173 026134R        JMP NXTBL    GO DO IT AGAIN
*
00174 002400  DONE   CLA          SET STATUS TO OK
00175 172004R        STA STAT,I
*
00176 102100  ERROR  STF 0        INTERRUPTS ON
00177 126005R        JMP RDATA,I  RETURN
00176         ERR07  EQU ERROR
00176         ERR12  EQU ERROR
*
00200 000001  D1     DEC 1
00201 000000  INIT   NOP
00202 000000  INTST  NOP
00203 000000  SSECT  NOP
00204 000000  CNTR   NOP
00205 000000  MXBM   NOP
00206 000000  SPB    NOP
00207 000000  SPP    NOP
00210 000000  OFST   NOP
*
```

```
0161* THE ORDER OF THE FOLLOWING 14 CARDS MUST NOT BE DISTURBED



 PAGE 0003 #01  RDATA SUBROUTINE

0162  00211 000212R LBUF  DEF *+1
0163  00212 000000  TIME  BSS 2
0164  00214 000000  ANGL1 NOP
0165  00215 000000        NOP
0166  00216 000000  FSTB# NOP
0167  00217 000000  FBANG NOP
0168  00220 000000  PING  NOP
0169  00221 000000  #BMS  NOP
0170  00222 000000  S     NOP
0171  00223 000000  M     NOP
0172  00224 000000  K     NOP
0173  00225 000000  N     NOP
0174  00226 000000  BLOCK NOP
0175  00227 000000        BSS 3         END OF SPECIAL BUFFER AREA
0176*
0177* RRERETRIEVE HEADER SUBROUTINE
0178*
0179  00232 000000  HPNG# NOP
0180  00233 000000  HDATA NOP
0181  00234 000000  HEADR NOP
0182  00235 016001X       JSB .ENTR
0183  00236 000232R       DEF HPNG#
0184  00237 002404        CLA,INA
0185  00240 102300        SFS 0         DETERMINE INT. SYS. STATUS
```

```
00241 002400           CLA
00242 072202R          STA INTST     INTST:  1=INT ON, 0=INT OFF
00243 103100           CLF 0         INTERRUPTS OFF
00244 062233R          LDA HDATA
00245 072300R          STA HLOC
00246 062201R          LDA INIT
00247 002002           SZA           POINTERS INITIALIZED? (=1)
00250 026261R          JMP RHDR      YES, SKIP READ OF 1ST HEADER
00251 002400           CLA           SET SPECIAL SECTOR POINTER TO
00252 072203R          STA SSECT       BEGINNING OF DATA
*
00253 016310R          JSB RDISC     READ PART OF FIRST SECTOR
00254 000257R          DEF *+3         OF DATA AREA
00255 000212R          DEF LBUF+1
00256 000020           DEC 16
00257 002404           CLA,INA       SET INIT=1 TO SKIP HEADER READ
00260 072201R          STA INIT        NEXT TIME
*
00261 062221R RHDR     LDA #BMS      COMPUTE # SECTORS PER BLOCK
00262 100200           MPY =D4         AND PING
00263 000555R
00264 072206R          STA SPB
00265 100200           MPY BLOCK     BEAMS X BLOCKS X 4 SECTORS/BEAM
00266 000226R
00267 002004           INA           INCLUDE 1 SECTOR FOR HEADER
00270 072207R          STA SPP
*
00271 162232R          LDA HPNG#,I
00272 042553R          ADA =D-1
00273 100200           MPY SPP       STARTING SECTOR FOR THIS PING
00274 000207R
00275 072203R          STA SSECT
*
00276 016310R          JSB RDISC     READ HEADER, WHICH IS PART OF
00277 000302R          DEF *+3         FIRST SECTOR OF EACH PING
```

```
0218  00300 000000  HLOC   NOP
0219  00301 000020         DEC 16
0220  00302 062202R        LDA INTST     GET INT. SYS. STATUS AT ENTRY
0221  00303 002002         SZA           IF ZERO, DO NOT TURN ON
0222  00304 102100         STF 0         INTERRUPTS ON
0223  00305 126234R        JMP HEADR,I   RETURN
0224*
0225* DISC SUBROUTINES
0226*
0227  00306 000000  RPTR   NOP
0228  00307 000000  RCTR   NOP
0229  00310 000000  RDISC  NOP
0230  00311 016001X        JSB .ENTR
0231  00312 000306R        DEF RPTR
0232*
0233  00313 062203R        LDA SSECT     CONVERT SSECT TO TRACK AND SECTOR
0234  00314 006400         CLB
0235  00315 100400         DIV =D48
      00316 000557R
0236  00317 072544R        STA STRAK
0237  00320 076543R        STB SECT
0238  00321 062560R        LDA =D2       SET LOWER DISC HEAD BASE
0239  00322 072546R        STA HEADB
0240  00323 062544R        LDA STRAK
0241  00324 066004X        LDB LTLD      ANY SPACE LOWER DISC?
0242  00325 076545R        STB MXTRK
0243  00326 006003         SZB,RSS
0244  00327 026337R        JMP RDIS3     NO, GO SET UP UPPER DISC
0245  00330 042003X        ADA FTLD      COMPUTE ACTUAL TRACK
0246  00331 072542R        STA TRAK
0247  00332 003004         CMA,INA
0248  00333 042004X        ADA LTLD
0249  00334 002021         SSA,RSS       GREATER THAN LAST TRACK?
0250  00335 026353R        JMP RDIS1     NO, GO READ
0251  00336 003000         CMA
```

```
0252  00337 066006X RDIS3 LDB LTUD      ANY SPACE UPPER DISC?
0253  00340 076545R       STB MXTRK
0254  00341 006003        SZB,RSS
0255  00342 026176R       JMP ERR07     NO, ERROR
0256  00343 006400        CLB           SET UPPER DISC HEAD BASE
0257  00344 076546R       STB HEADB
0258  00345 042005X       ADA FTUD      COMPUTE ACTUAL TRACK
0259  00346 072542R       STA TRAK
0260  00347 003004        CMA,INA
0261  00350 042006X       ADA LTUD
0262  00351 002020        SSA           TRACK GREATER THAN LAST TRACK?
0263  00352 026176R       JMP ERR07     YES, ERROR
0264*
0265  00353 062543R RDIS1 LDA SECT
0266  00354 100200        MPY =D128
      00355 000556R
0267  00356 003004        CMA,INA
0268  00357 042561R       ADA =D6144    AA = 6144 - SECT*128
0269  00360 072541R       STA AA        # WORDS REMAINING IN TRACK
0270  00361 066307R       LDB RCTR
0271  00362 003004        CMA,INA
0272  00363 042307R       ADA RCTR
0273  00364 002020        SSA           AA .GT. RCTR?
0274  00365 076541R       STB AA        YES, SUBSTITUTE RCTR FOR AA
0275  00366 016433R       JSB DISCR     READ AA WORDS
0276*
0277  00367 006400        CLB           UPDATE SECTOR
0278  00370 062541R       LDA AA        THIS ROUTINE USES AN ARTIFICIAL
0279  00371 100400        DIV =D128       SECTOR IN RANGE 0-47
      00372 000556R
0280  00373 042543R       ADA SECT
0281  00374 072543R       STA SECT      SECT = SECT + AA/128
0282*
0283  00375 052557R       CPA =D48
0284  00376 002001        RSS
```

```
00377 026421R       JMP RDIS2    SECTOR NOT 48, SKIP TRACK UPDATE
00400 002400        CLA
00401 072543R       STA SECT
00402 036542R       ISZ TRAK
00403 062542R       LDA TRAK
00404 003004        CMA,INA
00405 042545R       ADA MXTRK
00406 002021        SSA,RSS      TRAK .GT. MXTRK?
00407 026421R       JMP RDIS2    NO
00410 062005X       LDA FTUD     YES, GO TO SECOND DISC
00411 072542R       STA TRAK
00412 062006X       LDA LTUD
00413 072545R       STA MXTRK
00414 062546R       LDA HEADB
00415 002003        SZA,RSS      ALREADY ON UPPER DISC?
00416 026176R       JMP ERR07    ERROR, STORAGE EXCEEDED
00417 002400        CLA
00420 072546R       STA HEADB    HEAD BASE = 1ST UPPR DISC HEAD
*
00421 062541R RDIS2 LDA AA       UPDATE POINTER AND COUNTER
00422 042306R       ADA RPTR
00423 072306R       STA RPTR
00424 062541R       LDA AA
00425 003004        CMA,INA
00426 042307R       ADA RCTR
00427 072307R       STA RCTR
00430 002002        SZA          RCTR=0?
00431 026353R       JMP RDIS1    NO, GO DO ANOTHER READ
00432 126310R       JMP RDISC,I  RETURN
*
00433 000000  DISCR NOP
00434 062543R       LDA SECT     CONVERT SECT TO HEAD AND SECTOR
00435 070001        STA B
00436 042562R       ADA =D-24
00437 002020        SSA
```

```
0320  00440 026445R        JMP FRSTH
```



```
0321  00441 070001         STA B
0322  00442 062546R        LDA HEADB
0323  00443 002004         INA
0324  00444 002001         RSS
0325  00445 062546R FRSTH  LDA HEADB
0326  00446 001727         ALF,ALF
0327  00447 030001         IOR B
0328  00450 072547R        STA HDSEC
0329  00451 016501R        JSB SEEK
0330  00452 106717         CLC DISCD
0331  00453 106720         CLC DISCC
0332  00454 062563R        LDA =B120000 SET UP DMA
0333  00455 032552R        IOR SELDD
0334  00456 102607         OTA 7
0335  00457 106703         CLC 3
0336  00460 062306R        LDA RPTR
0337  00461 032564R        IOR =B100000
0338  00462 102503         OTA 3
0339  00463 102703         STC 3
0340  00464 062541R        LDA AA         # WORDS TO READ
0341  00465 003004         CMA,INA
0342  00466 102603         OTA 3
0343  00467 062550R        LDA RDCMD
0344  00470 102620         OTA DISCC
```

```
00471 103717        STC DISCD,C
00472 103707        STC 7,C
00473 103720        STC DISCC,C
00474 102320        SFS DISCC     WAIT FOR COMPLETION
00475 026474R       JMP *-1
00476 106707        CLC 7         STOP DMA
00477 016523R       JSB STATS     CHECK STATUS
00500 126433R       JMP DISCR,I   RETURN
*
00501 000000  SEEK  NOP
00502 106717        CLC DISCD
00503 106720        CLC DISCC
00504 062542R       LDA TRAK
00505 102617        OTA DISCD
00506 103717        STC DISCD,C
00507 062551R       LDA SKCMD
00510 102620        OTA DISCC
00511 103720        STC DISCC,C
00512 102317        SFS DISCD
00513 026512R       JMP *-1
00514 062547R       LDA HDSEC
00515 102617        OTA DISCD
00516 103717        STC DISCD,C
00517 102320        SFS DISCC
00520 026517R       JMP *-1
00521 016523R       JSB STATS
00522 126501R       JMP SEEK,I
*
00523 000000  STATS NOP
00524 106717        CLC DISCD
00525 106720        CLC DISCC
00526 002400        CLA           STATUS COMMAND
00527 102620        OTA DISCC
00530 103720        STC DISCC,C
00531 102317        SFS DISCD
```

```
0380  00532 026531R        JMP *-1
0381  00533 102517         LIA DISCD
0382  00534 000010         SLA
0383  00535 026537R        JMP ERR10     ERROR, DISC STATUS BAD
0384  00536 126523R        JMP STATS,I   RETURN
0385*
0386  00537 102010  ERR10 HLT 10B        DISC STATUS BAD:
0387  00540 026537R        JMP *-1         IRRECOVERABLE ERROR
0388*
0389  00541 000000  AA    NOP
0390  00542 000000  TRAK  NOP
0391  00543 000000  SECT  NOP
0392  00544 000000  STRAK NOP
0393  00545 000000  MXTRK NOP
0394  00546 000000  HEADB NOP
0395  00547 000000  HDSEC NOP
0396  00550 020000  RDCMD OCT 20000
0397  00551 030000  SKCMD OCT 30000
0398  00552 000017  SELDD DEF DISCD
0399  00017         DISCD EQU 17B
0400  00020         DISCC EQU 20B
0401  00000         A     EQU 0
0402  00001         B     EQU 1
      00553 177777
      00554 020020
      00555 020004
      00556 020200
      00557 000060
      00560 000002
      00561 014000
      00562 177750
      00563 120000
      00564 100000
0403                      END
**  NO ERRORS *TOTAL **RTE ASMB 750420**
```

TRACK ALLOCATION TABLE - TAT

1. FUNCTION. TAT is a non-executable program unit which provides the disc track availability information to all programs.

2. CONSTRAINTS. None.

3. CALLING SEQUENCE. Called by: N/A; used by: Y0001, RDATA.

4. DESCRIPTION OF INPUT. N/A

5. DESCRIPTION OF OUTPUT. N/A

6. FILES USED. N/A

7. ERRORS AND RESTRICTIONS. N/A

8. COMPUTER OPERATOR INSTRUCTIONS. N/A

9. DESCRIPTION OF PROCESSING. TAT has the following four entry points. Referencing them as externals provides the required information to the other program.

FTLD - first track lower disc

LTLD - last track lower disc

FTUD - first track upper disc

LTUD - last track upper disc

TAT1    T=00004 IS ON CR00001 USING 00003 BLKS R=0014

```
ASMB,R,L
      HED TRACK ALLOCATION TABLE FOR Y0001 AND RDATA
      NAM TAT
      ENT FTLD,LTLD,FTUD,LTUD
*

* WHENEVER THIS TABLE IS CHANGED, FILES #TAT1 AND #DCP MUST BE
* UPDATED AND THE DATA COLLECTION AND ANALYSIS PROGRAMS MUST
* BE REBUILT.  IT IS IMPORTANT THAT BOTH SYSTEMS USE THE SAME
* TAT.
FTLD  DEC 0
LTLD  DEC 0
FTUD  DEC 61
LTUD  DEC 202
      END
```

:

```
TEKS    T=00003 IS ON CR00001 USING 00056 BLKS R=0000

ASMB,B,L,R
      NAM PLOTP
      ENT CHIN
      ENT CHOUT
      ENT CURSI
      ENT TPLOT
*
A     EQU 0          A REGISTER  DEFINITION
B     EQU 1          B REGISTER  DEFINITION
IMPUR EQU 0          TAGS ALTERABLE LOCATIONS
**                                                   *
SC    EQU 11B        SELECT CODE OF INTERFACE
US    OCT 37         CHAR TO SET ALPHA MODE
GS    OCT 35         CHAR TO SET VECTOR MODE
ESC   OCT 33         ESC+SUB ENABLE GRAPHICS CURSOR
SUB   OCT 32         CHAR TO ENABLE GRAPHICS CURSOR
*
M2    DEC -2
B17   OCT 17
B37   OCT 37
B40   OCT 40
HOB   OCT 40
LXB   OCT 100
LYB   OCT 140
B377  OCT 377
BIT15 OCT 100000
TYPB  OCT 120000     TTY OUTPUT BITS
TYRB  OCT 160000     TTY INPUT BITS
ARGAD BSS 1
RETAD BSS 1
X     BSS 1
Y     BSS 1
```

```
SAVEX  BSS 1          SAVE LOW ORDER X BITS
PPSW   OCT 0          POINT PLOT MODE SWITCH
       HED CHOUT
*
CHOUT  ABS IMFUR
       JSB LENTR      SET UP THE POINTERS
       LDA ARGAD,I    GET IDATA
       JSB LCHOT        AND OUTPUT IT
       JMP RETAD,I    RETURN
       HED CHIN
CHIN   ABS IMFUR
       JSB LENTR      SET UP THE POINTERS
       JSB LCHIN      GET AN INPUT CHARACTER
       STA ARGAD,I    STORE IT INTO IDATA
       JMP RETAD,I    RETURN
       HED TPLOT
TPLOT  ABS IMFUR
       JSB LENTR      SET UP POINTERS
       LDA ARGAD,I    GET THE FIRST ARGUEMENT
       ISZ ARGAD      ADVANCE ARGUEMENT POINTER
       SZA,RSS        CHECK FOR CASE 1, I = 0
       JMP DARK
       SSA            SKIP IF BRIGHT VECTOR
       JMP POINT      CASE 3, I < 0
       JMP POIBR      CASE 2, I > 0
       HED TPLOT  CASE 1
*
*
DARK   EQU *
       LDA GS         GET THE CODE FOR VECTOR MODE
       JMP POIFE        AND ENTER POINT PLOT
       HED TPLOT  CASE 3
*
*
POINT  EQU *
```

```
        LDA GS        GET THE CODE FOR POINT PLOT
        STA PPSW      SET POINT PLOT SWITCH
POIFE   EQU *
        JSB LCHOT       AND OUTPUT IT
POIBR   EQU *
        LDA ARGAD,I   GET X
        STA X           AND SAVE IT LOCALLY
        ISZ ARGAD     ADVANCE TO NEXT ARGUEMENT
        LDA ARGAD,I   GET Y
        STA Y
        JSB PLOT      NOW PLOT THIS POINT
        LDA SAVEX     LOAD LOW ORDER X
        LDB PPSW      LOAD POINT PLOT SWITCH
        SZB           SKIP IF NOT POINT PLOT MODE
        JSB LCHOT     OUTPUT LOW ORDER X
        CLA
        STA PPSW      RESET POINT PLOT SWITCH
        JMP RETAD,I   RETURN
        HED CURSI
CURSI   ABS IMFUR
        JSB LENTR     SET UP THE POINTERS
        LDA US        GET THE CODE FOR ALPHA MODE
        JSB LCHOT       AND OUTPUT IT
        LDA ESC       GET THE CODE TO ENABLE GRAPHIC INPUT
        JSB LCHOT       AND OUTPUT IT
        LDA SUB
        JSB LCHOT
*
        JSB LCHIN     WAIT FOR THE FIRST INPUT
        STA ARGAD,I   SAVE THE CHARACTER
        ISZ ARGAD     ADVANCE POINTER TO NEXT ARGUEMENT
*
        JSB LCHIN     GET THE NEXT INPUT CHARACTER
        AND B37       MASK IT TO ONLY FIVE BITS
        ALF,RAL       POSITION FIVE BITS OVER
```

```
         STA X          AND SAVE THEM IN X
         JSB LCHIN    GET THE NEXT CHARACTER
         AND B37        AND MASK TO FIVE BITS
         IOR X        PLUG IN MSB
         STA ARGAD,I  AND RETURN X VALUE TO CALLER
         ISZ ARGAD    => Y ARGUEMENT
         JSB LCHIN    FETCH THE NEXT CHARACTER
         AND B37        AND MASK IT TO FIVE BITS
         ALF,RAL      MOVE FIVE PLACES TO THE LEFT
         STA Y          AND SAVE THEM IN Y
         JSB LCHIN    GET THE LAST CHARACTER
         AND B37        AND MASK IT OFF TO ONLY FIVE BITS
         IOR Y        PLUG IN MSB
         STA ARGAD,I  AND RETURN VALUE TO CALLER
         JMP RETAD,I  RETURN
         HED PLOT
*
PLOT     ABS IMFUR
         LDA Y        GET THE Y VALUE
         ALF,ALF      DISCARD THE RIGHTMOST
         ALF,RAR        FIVE BITS
         AND B37      MASK TO ONLY FIVE BITS
         IOR HOB      PLUG IN REQUIRED BITS
         JSB LCHOT      AND OUTPUT THE CHARACTER
         LDA Y        GET BACK THE Y VALUE
         AND B37        AND MASK TO ONLY FIVE BITS
         IOR LYB      PLUG IN REQUIRED BITS
         JSB LCHOT      AND OUTPUT THE CHARACTER
         LDA X        NOW GET THE X VALUE
         ALF,ALF      DISCARD THE RIGHTMOST
         ALF,RAR        FIVE BITS
         AND B37
         IOR HOB      PLUG IN THE REQUIRED BITS
         JSB LCHOT      AND OUTPUT THE CHARACTER
         LDA X        GET BACK THE X VALUE
```

```
      AND B37        AND LEAVE ONLY LOW ORDER FIVE
      IOR LXB      PLUG IN THE REQUIRED BITS
      STA SAVEX    SAVE LOW ORDER X BITS
      JSB LCHOT      AND OUTPUT THE CHARACTER
      JMP PLOT,I   RETURN
      HED LCHIN
LCHIN ABS IMFUR
LCHIP EQU *
      SFS SC       WE ARE NOT ALLOWED TO
      JMP LCHIP      TOUCH IF BEING USED
      LDA *        TIME WASTER
      LDA *
      SFS SC       CHECK ONCE MORE TO MAKE SURE
      JMP LCHIP
      LDA TYRB     GET THE TTY INPUT BITS
      OTA SC         AND OUTPUT TO THE INTERFACE
      STC SC,C     START THE INPUT
      CLC SC       PREVENT AN INTERRUPT
      SFS SC       WAIT FOR SOME ACTION
      JMP *-1
      LIA SC       GET THE CHARACTER
      AND B377       AND MASK IT OFF
      JMP LCHIN,I  RETURN
      HED LCHOT
LCHOT ABS IMFUR
      AND B377     MASK THE CHARACTER TO SIZE
LCHOP EQU *
      SFS SC       WE ARE NOT ALLOWED TO
      JMP LCHOP      TOUCH IF BEING USED
      LDB *        TIME WASTER
      LDB *
      SFS SC       CHECK ONCE MORE
      JMP LCHOP
      LDB TYPB     GET THE TTY OUTPUT BITS
      OTB SC         AND TELL INTERFACE BOARD
```

```
        OTA SC        PUT THE CHARACTER OUT
        STC SC,C      START THE OUTPUT
        CLC SC        PREVENT AN INTERRUPT
        SFS SC        WAIT FOR THE OUTPUT
        JMP *-1         TO FINISH
        JMP LCHOT,I   RETURN
        HED LENTR
LENTR   ABS IMPUR
        LDA LENTR     A = ADDRESS OF OUR CALLER + 2
        ADA M2        SUBTRACT OFF THE TWO
        LDA A,I       FIND WHERE HE WAS CALLED FROM
        IOR BIT15     PLUG IN THE INDIRECT BIT
        STA RETAD     SAVE PTR TO RETURN ADDRESS
        INA           ADVANCE POINTER TO ARGUEMENT LIST
        STA ARGAD       AND SAVE FOR THE ROUTINE
        JMP LENTR,I   RETURN
        END
**END-OF-TAPE
*

:
```

KEYBOARD INPUT AND SPECIAL LINE PRINTING - TYPI, TYPA, TYPC, PNTL

1. FUNCTION. TYPI, TYPA, and TYPC perform keyboard input of integer strings and multiple ASCII characters, respectively, bypassing IOC and its requirement for line feed to be pressed and its lack of error checking. They also permit null entries. PNTL outputs a line of ASCII characters based on the contents of the printer buffer, with the buffer length given.

2. CONSTRAINTS. A maximum of 20 characters may be entered. No check is made for multiple + or - signs or values greater than can be held in 16 bits all of which are error conditions.

3. CALLING SEQUENCE. Called by: Y1002, Y0003, INPUT, ASKQ

CALL TYPI (INVAL,IFLG)

where
IVAL is the location where the entered value will be stored
IFLG is the location where the error flag will be stored.

CALL TYPA (IVAL,IFLG)

where
IVAL is the location where the ASCII character will be stored
IFLG is the location where the error flag will be stored
IFLG is set negative for error, zero for no input, and positive for good input.

TYPC - performs keyboard input of up to ten (1Ø) ASCII characters.

CALL TYPC (AC,IFLGC)

AC: Location of buffer where the ASCII characters are stored.

IFLGC: Location where error flag is stored. (IFLGC values same as TYPA'S IFLGA values)

PNTL - prints one line of formatted data without doing a carriage return and line feed.

CALL PNTL (IPNTB,IBLG)

IPNTB: print buffer address

IBLG: buffer length

4. DESCRIPTION OF PROCESSING. Input is an ASCII string entered through the keyboard, followed by a carriage return. A DEL character deletes the entire line. A carriage return alone causes a return with IFLG set to zero. PNTL input is the contents of the printer buffer (IPNTB).

5. DESCRIPTION OF OUTPUT. Output is the value entered in integer format, or the single ASCII character (for TYPA). TYPC allows for ten ASCII characters to be output.

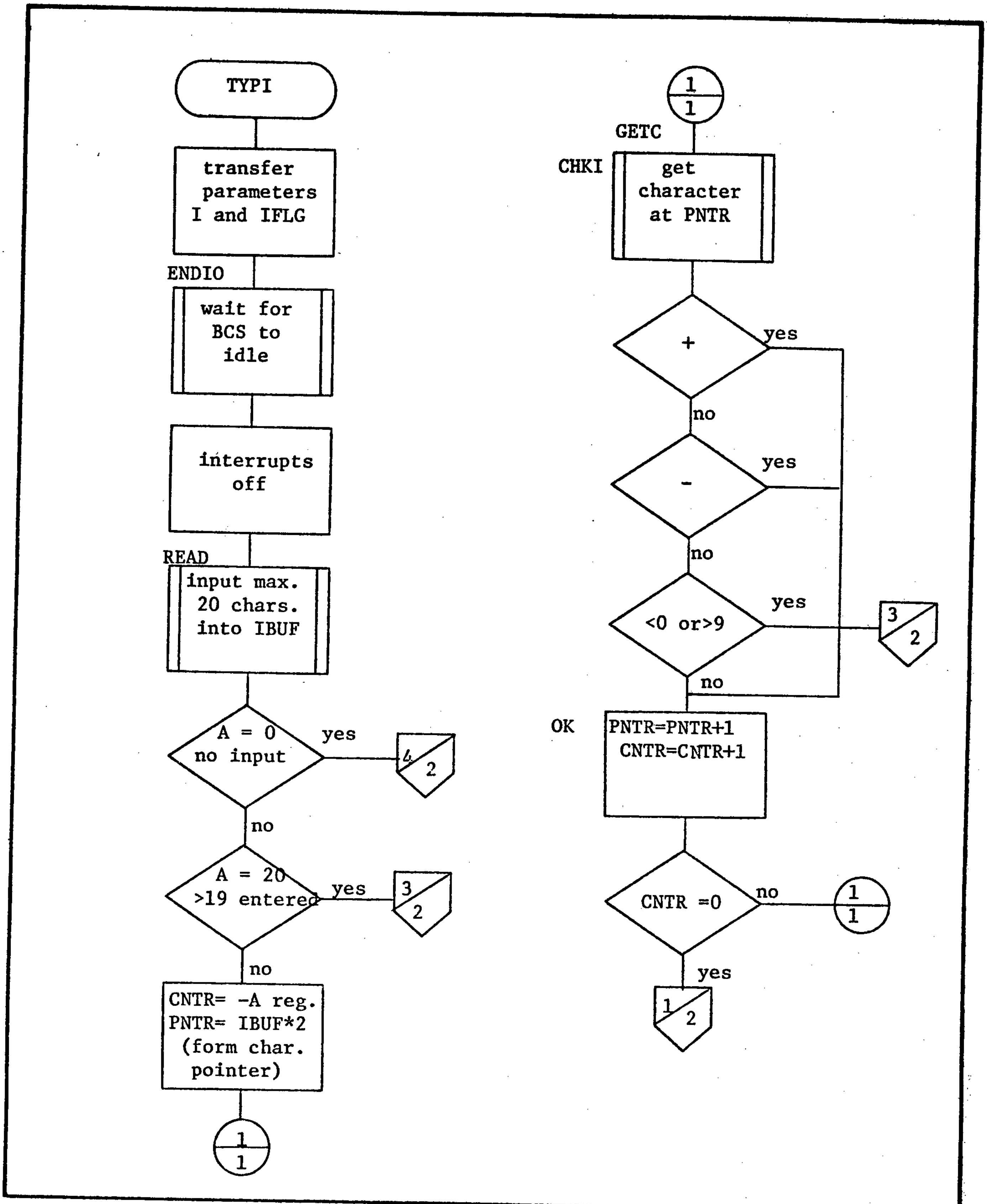
6. FILES USED. N/A

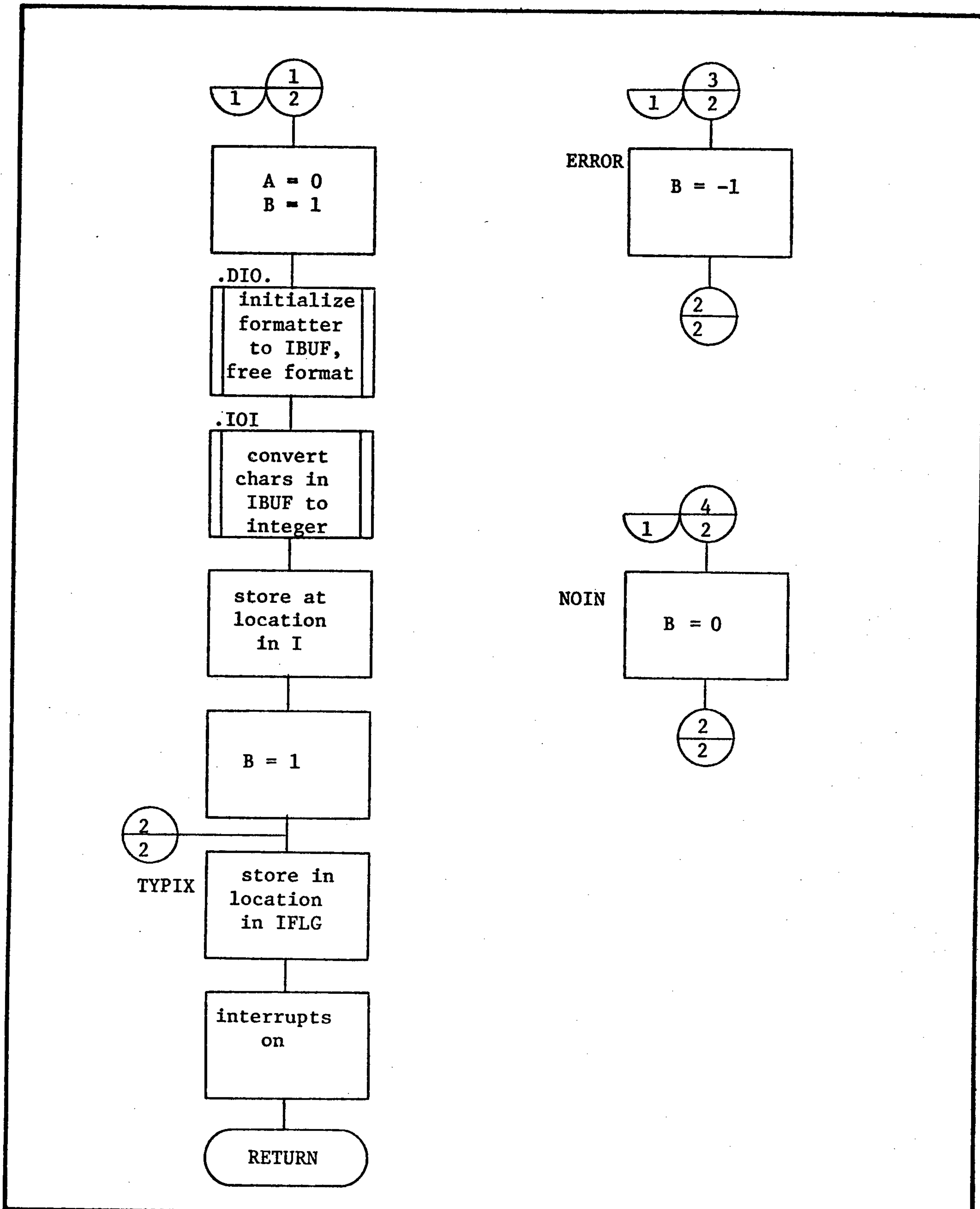
7. ERRORS AND RESTRICTIONS. None.

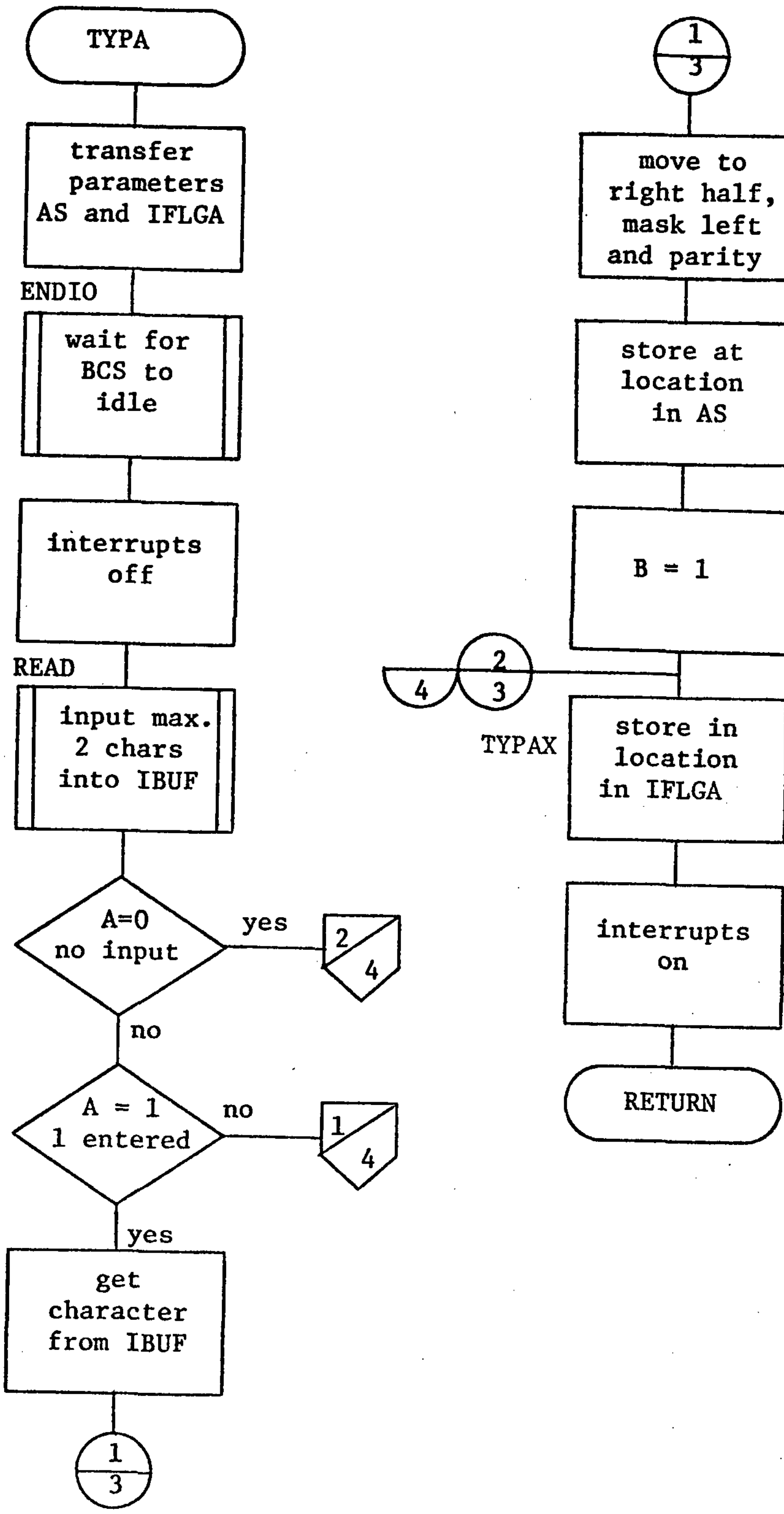
8. COMPUTER OPERATOR INSTRUCTIONS. N/A.

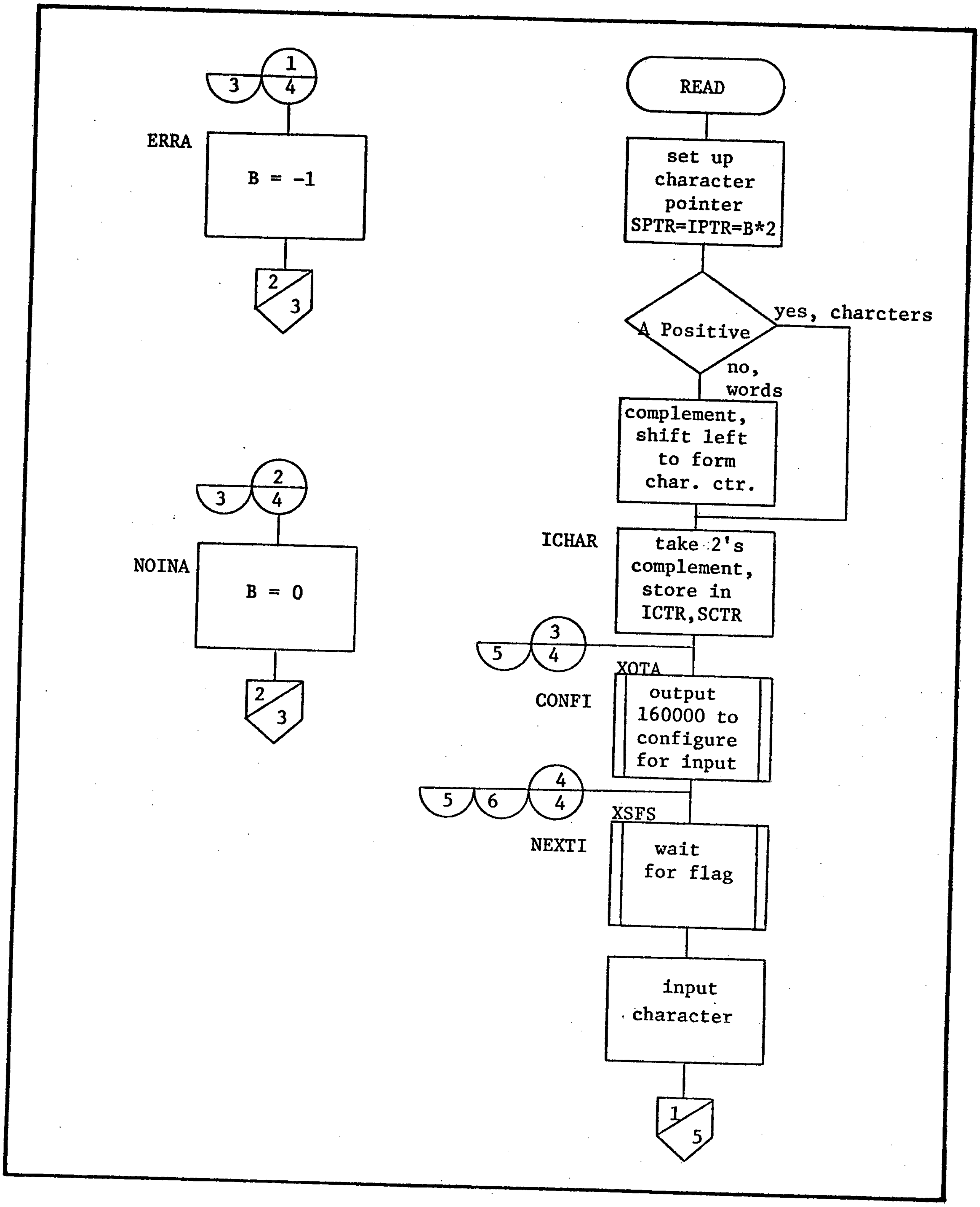
9. DESCRIPTION OF PROCESSING. TYPI, TYPA, and TYPC use their own I/O routines. To keep from interrupting BCS output, ENDIO is called at the beginning of each. The Formatter is called to convert the integer character string to binary in its internal conversion mode. However, in this mode there is no capability for supplying a character count. To overcome this shortcoming, a space is stored following the last character, as a delimiter. All other program details are shown in the flowchart, and the user is directed there for further information.

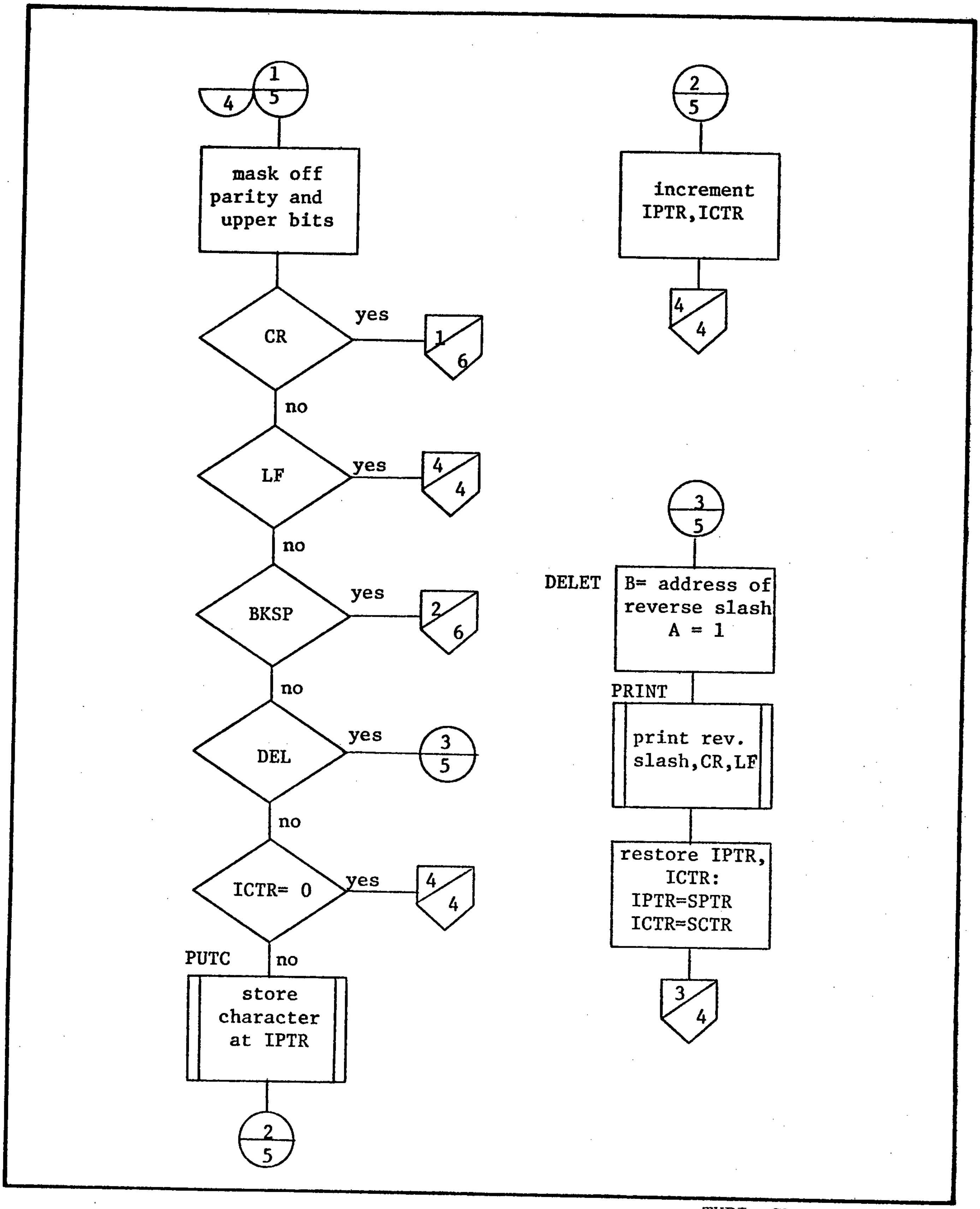
PNTL prints one line of formatted data without doing a carriage return and line feed. PNTL uses its own I/O routines. To keep from interrupting BCS output, ENDIO is called at the beginning of the routine. The pointer to the printer buffer is set into PNTR with the buffer length set into IBLG. The characters are then output. All other program details are shown in the flowchart and the user is directed there for further information.

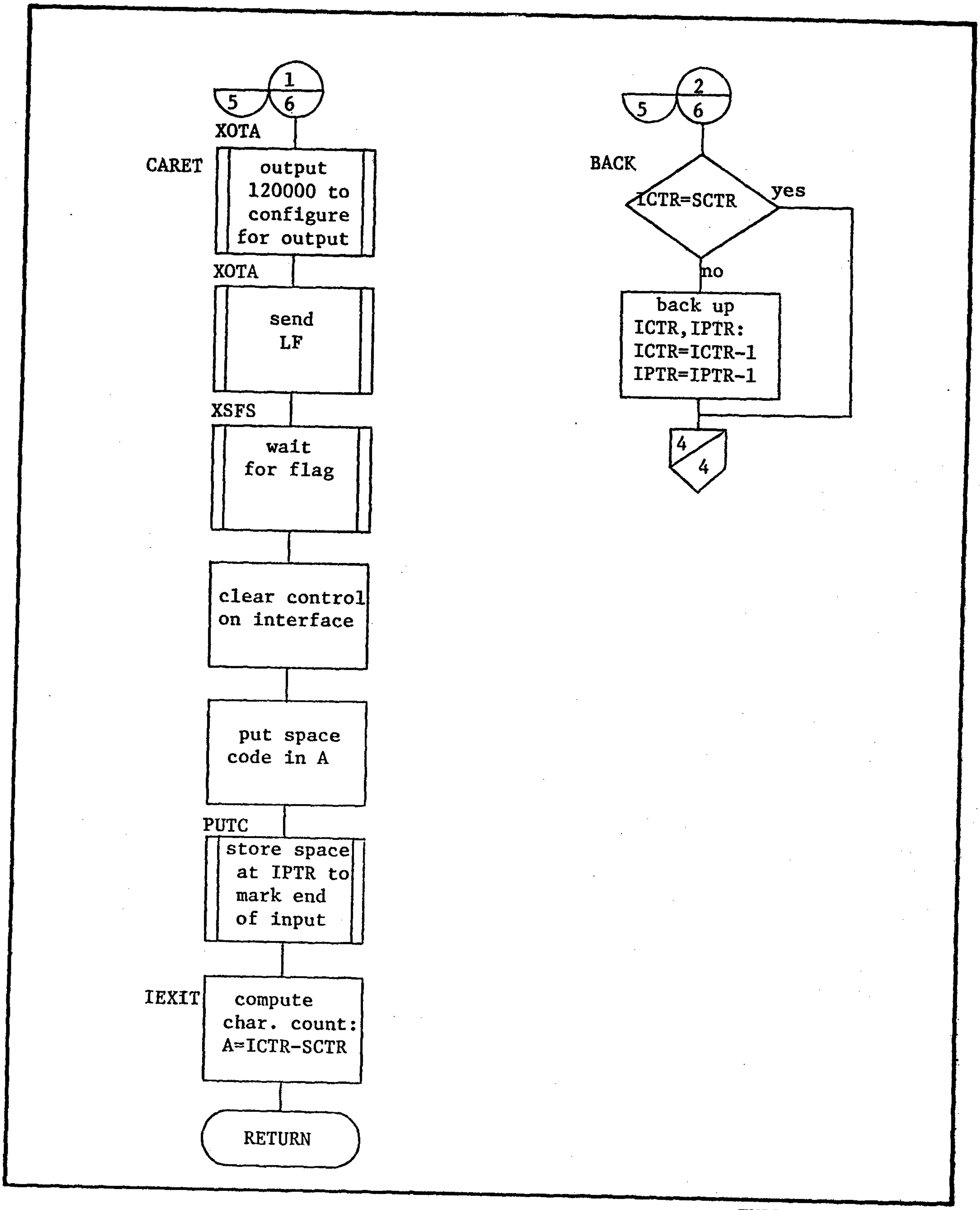

TYPI
transfer parameters I and IFLG
ENDIO
wait for BCS to idle
interrupts off
READ
input max. 20 chars. into IBUF
A = 0 no input
yes
4 2
no
A = 20 >19 entered
yes
3 2
no
CNTR= -A reg. PNTR= IBUF*2 (form char. pointer)
1 1
1 1
GETC
CHKI
get character at PNTR
+
yes
no
-
yes
no
<0 or>9
yes
3 2
no
OK
PNTR=PNTR+1 CNTR=CNTR+1
CNTR =0
no
1 1
yes
1 2

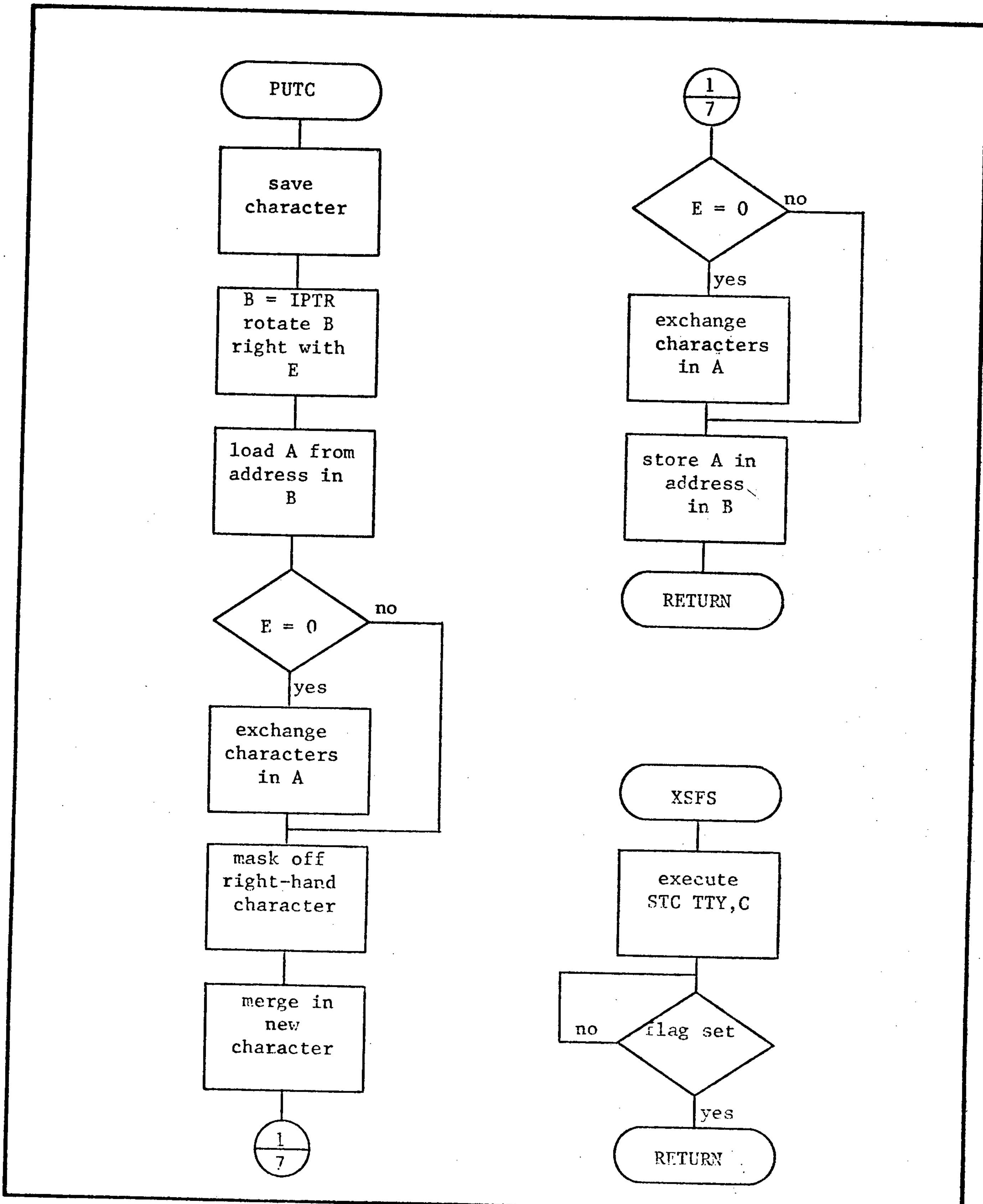

1
1
2
A = 0
B = 1
.DIO.
initialize formatter to IBUF, free format
.IOI
convert chars in IBUF to integer
store at location in I
B = 1
2
2
TYPIX
store in location in IFLG
interrupts on
RETURN
1
3
2
ERROR
B = -1
2
2
1
4
2
NOIN
B = 0
2
2

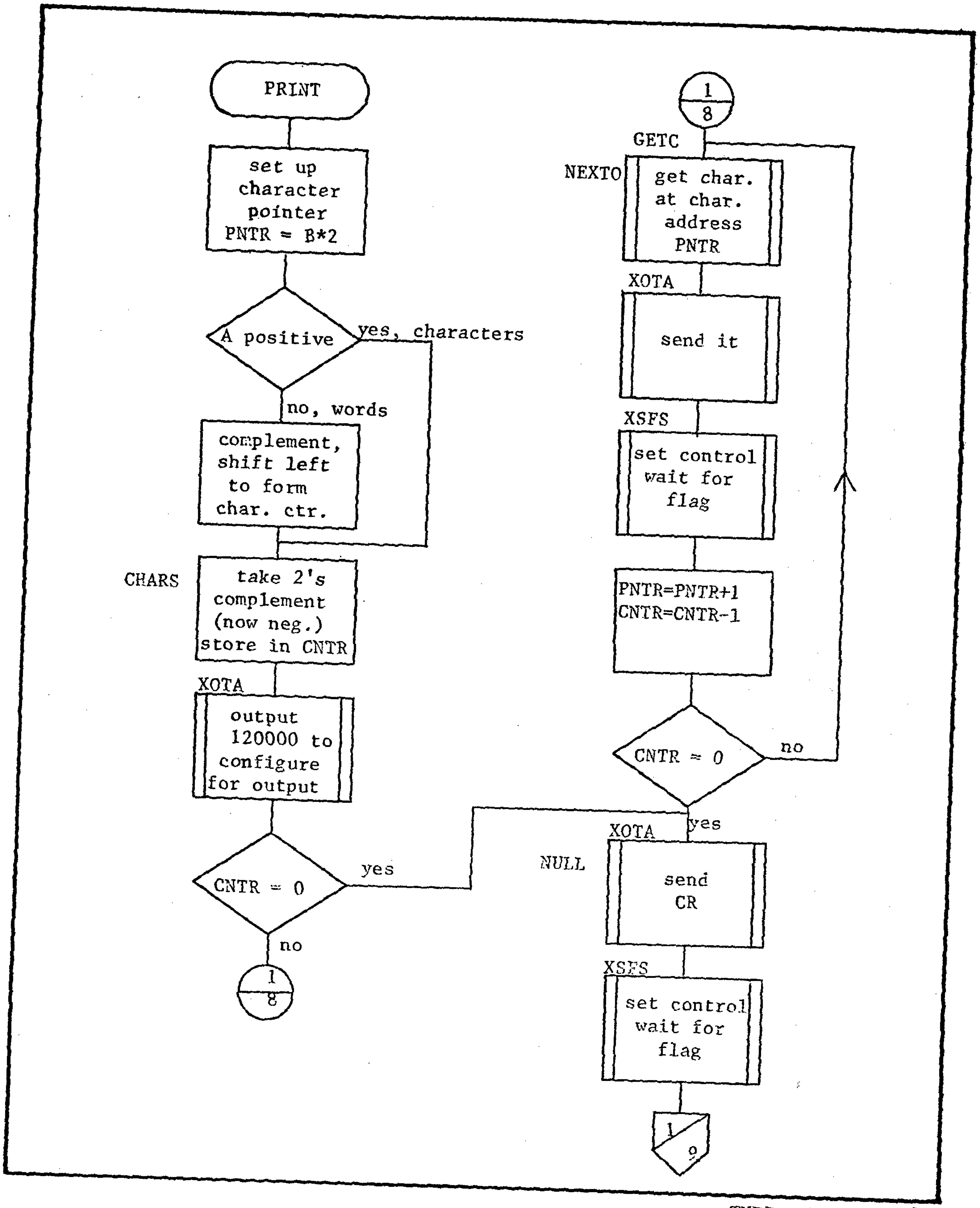

TYPA
transfer parameters AS and IFLGA
ENDIO
wait for BCS to idle
interrupts off
READ
input max. 2 chars into IBUF
A=0 no input
yes
2
4
no
A = 1 1 entered
no
1
4
yes
get character from IBUF
1
3
1
3
move to right half, mask left and parity
store at location in AS
B = 1
4
2
3
TYPAX
store in location in IFLGA
interrupts on
RETURN

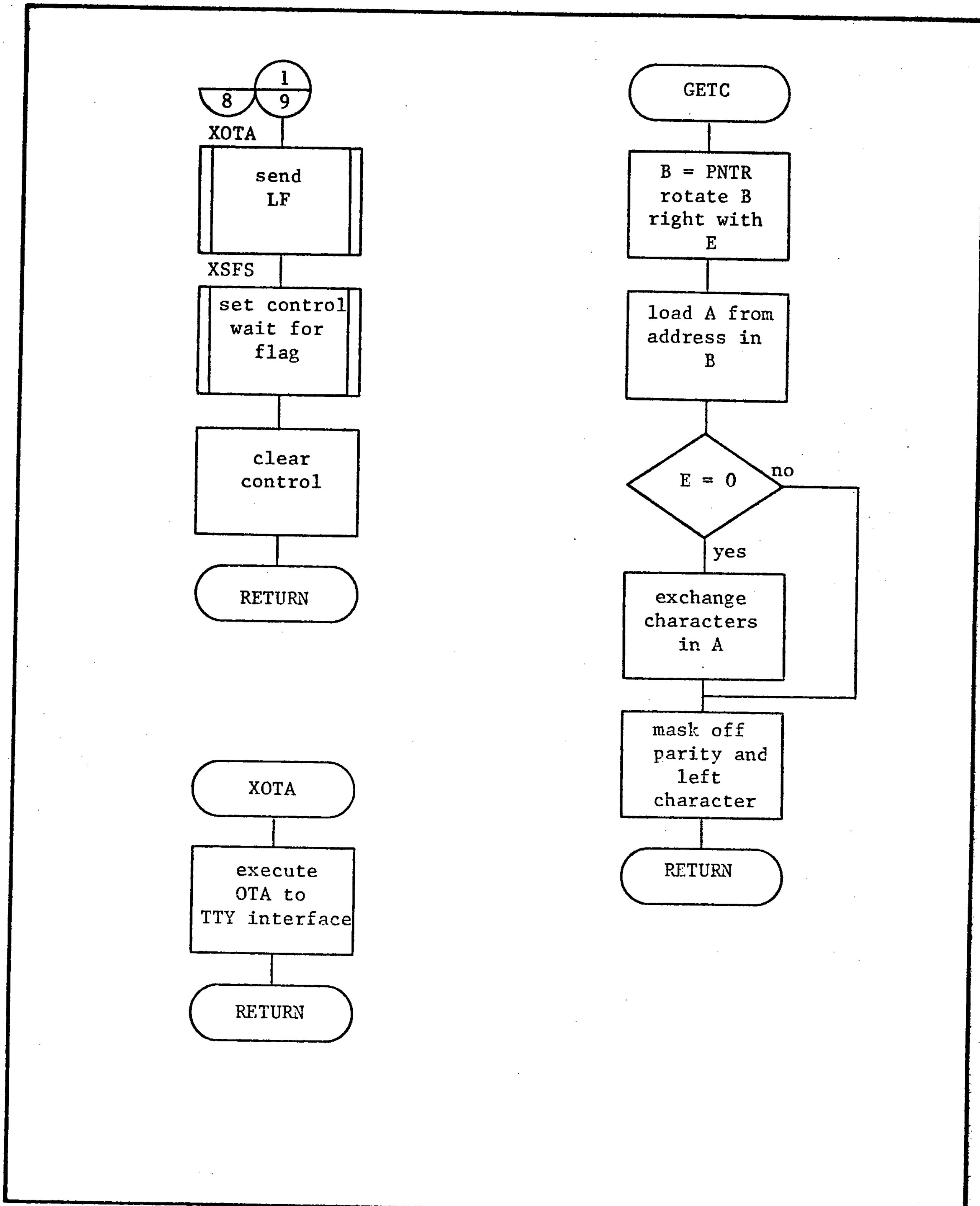

1
3
4
ERRA
B = -1
2
3
2
3
4
NOINA
B = 0
2
3
READ
set up character pointer SPTR=IPTR=B*2
A Positive
yes, charcters
no, words
complement, shift left to form char. ctr.
ICHAR
take 2's complement, store in ICTR,SCTR
3
5
4
XOTA
CONFI
output 160000 to configure for input
4
5
6
4
XSFS
NEXTI
wait for flag
input character
1
5


1/5
4
mask off parity and upper bits
CR
yes
1/6
no
LF
yes
4/4
no
BKSP
yes
2/6
no
DEL
yes
3/5
no
ICTR= 0
yes
4/4
PUTC
no
store character at IPTR
2/5
2/5
increment IPTR,ICTR
4/4
3/5
DELET
B= address of reverse slash A = 1
PRINT
print rev. slash,CR,LF
restore IPTR, ICTR: IPTR=SPTR ICTR=SCTR
3/4

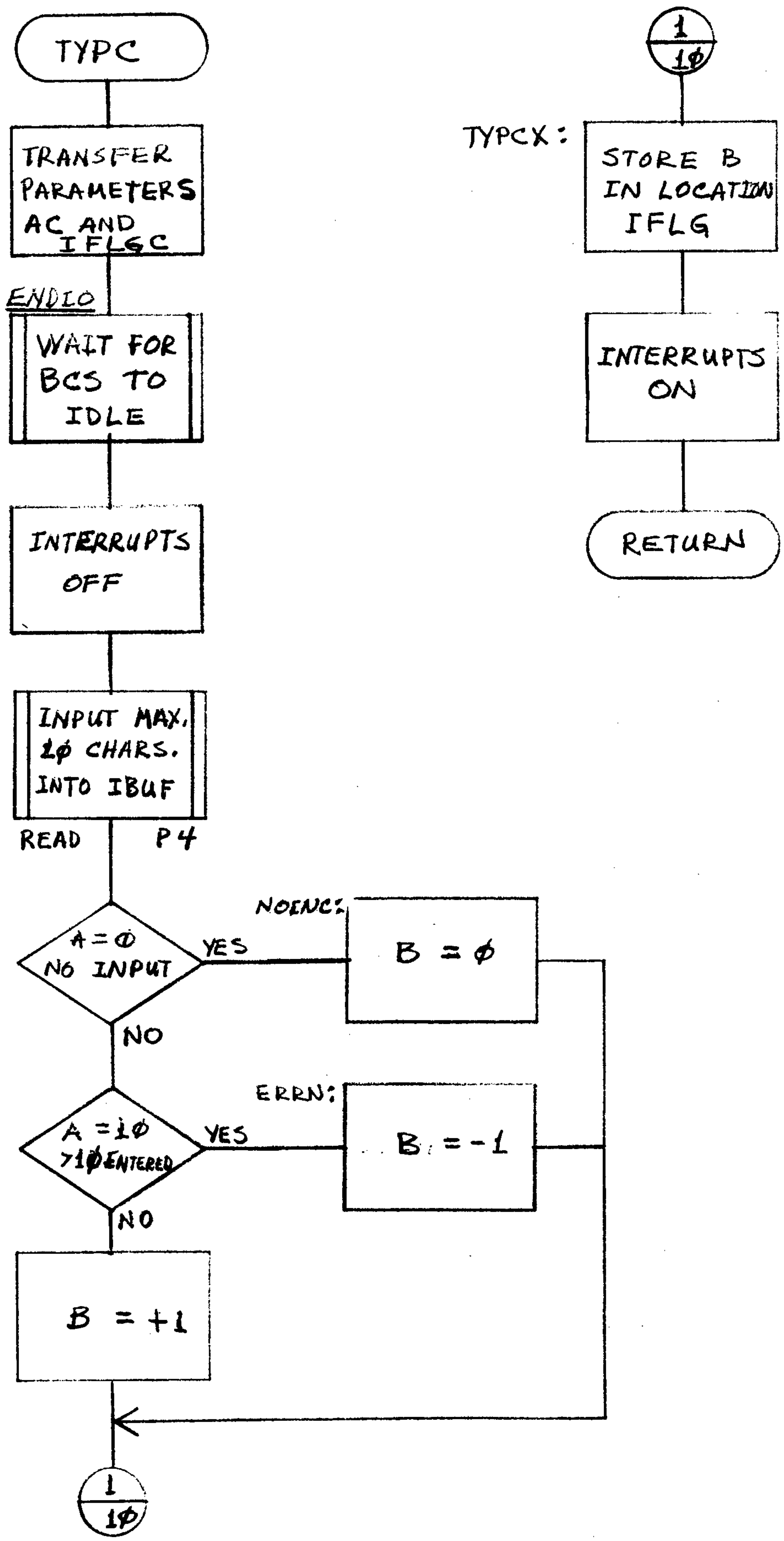

1
5
6
XOTA
CARET
output 120000 to configure for output
XOTA
send LF
XSFS
wait for flag
clear control on interface
put space code in A
PUTC
store space at IPTR to mark end of input
IEXIT
compute char. count: A=ICTR-SCTR
RETURN
2
5
6
BACK
ICTR=SCTR
yes
no
back up ICTR,IPTR: ICTR=ICTR-1 IPTR=IPTR-1
4
4

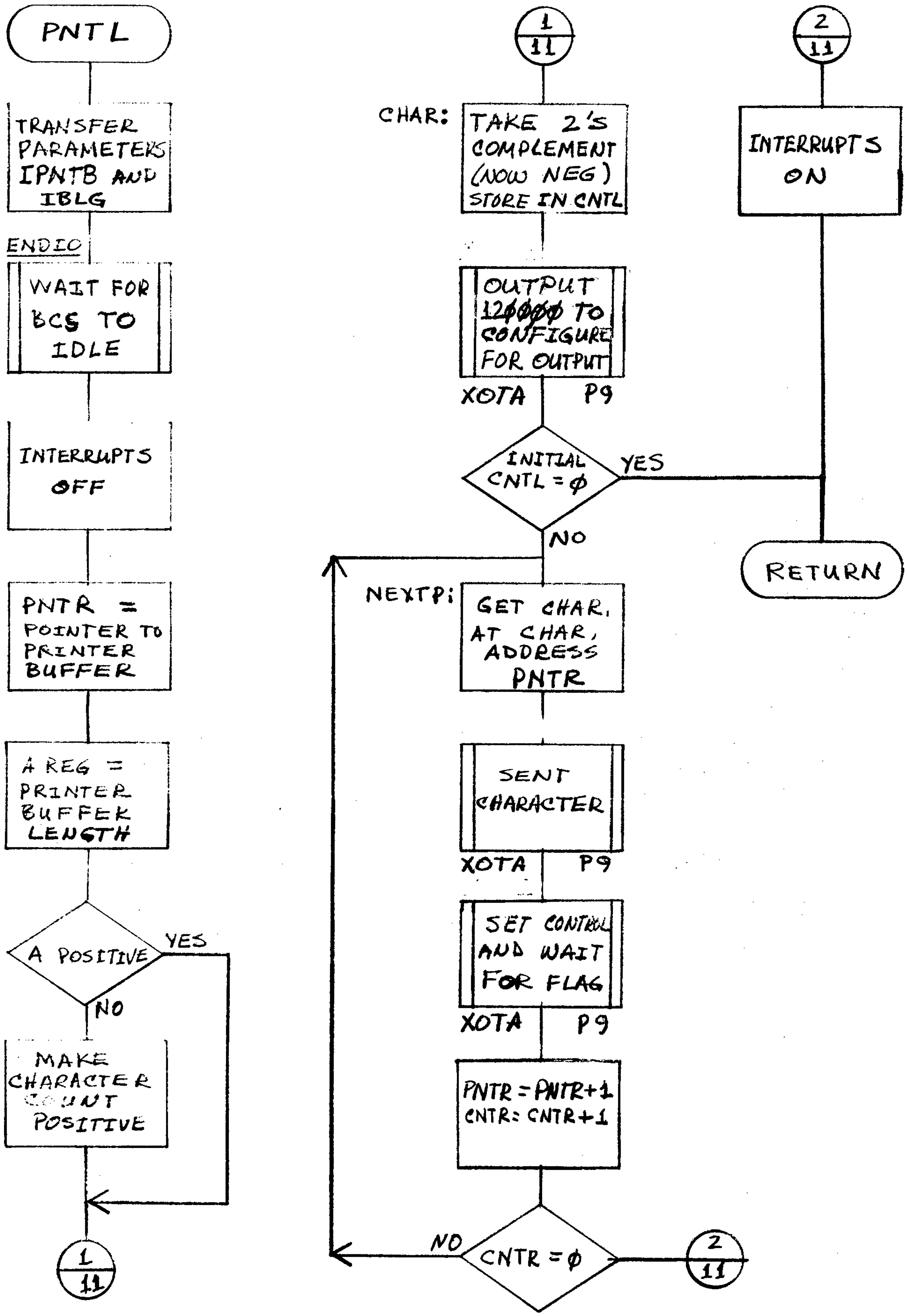

PUTC
save character
B = IPTR rotate B right with E
load A from address in B
E = 0
no
yes
exchange characters in A
mask off right-hand character
merge in new character
1/7
1/7
E = 0
no
yes
exchange characters in A
store A in address in B
RETURN
XSFS
execute STC TTY,C
flag set
no
yes
RETURN


PRINT
set up character pointer PNTR = B*2
A positive
yes, characters
no, words
complement, shift left to form char. ctr.
CHARS
take 2's complement (now neg.) store in CNTR
XOTA
output 120000 to configure for output
CNTR = 0
yes
no
1
8
GETC
NEXTO
get char. at char. address PNTR
XOTA
send it
XSFS
set control wait for flag
PNTR=PNTR+1
CNTR=CNTR-1
CNTR = 0
no
yes
XOTA
NULL
send CR
XSFS
set control wait for flag
1
9



1
8
9
XOTA
send LF
XSFS
set control wait for flag
clear control
RETURN
XOTA
execute OTA to TTY interface
RETURN
GETC
B = PNTR rotate B right with E
load A from address in B
E = 0
no
yes
exchange characters in A
mask off parity and left character
RETURN


TYPC
TRANSFER PARAMETERS AC AND IFLGC
ENDIO
WAIT FOR BCS TO IDLE
INTERRUPTS OFF
INPUT MAX. 1Ø CHARS. INTO IBUF
READ
P4
A = Ø NO INPUT
YES
NOINC:
B = Ø
NO
A = 1Ø >1Ø ENTERED
YES
ERRN:
B = -1
NO
B = +1
1
1Ø
1
1Ø
TYPCX:
STORE B IN LOCATION IFLG
INTERRUPTS ON
RETURN


PNTL
TRANSFER PARAMETERS IPNTB AND IBLG
ENDIO
WAIT FOR BCS TO IDLE
INTERRUPTS OFF
PNTR = POINTER TO PRINTER BUFFER
A REG = PRINTER BUFFER LENGTH
A POSITIVE
YES
NO
MAKE CHARACTER COUNT POSITIVE
1/11
CHAR:
TAKE 2'S COMPLEMENT (NOW NEG) STORE IN CNTL
OUTPUT 120000 TO CONFIGURE FOR OUTPUT
XOTA P9
INITIAL CNTL = 0
YES
NO
NEXTP:
GET CHAR. AT CHAR. ADDRESS PNTR
SENT CHARACTER
XOTA P9
SET CONTROL AND WAIT FOR FLAG
XOTA P9
PNTR = PNTR+1
CNTR = CNTR+1
CNTR = 0
NO
2/11
INTERRUPTS ON
RETURN

```
0001                    ASMB,R,L
0003  00000                   NAM TYPI1
0004                          ENT TYPI,TYPA,TYPC,PNTL
0005                          EXT .ENTR,.DIO.,.IOI.,ENDIO
0006*
0007*  ***************************************************************
0008*  *                                                             *
0009*  *  TITLE:    KEYBOARD INPUT AND SPECIAL LINE PRINTING         *
0010*  *                                                             *
0011*  *  MODULE:   TYPI        SOURCE:  TYPI1      VERSION:  X00    *
0012*  *                        OBJECT:  #TYPI1                      *
0013*  *                                                             *
0014*  *  AUTHOR:   R. TASHJIAN          DATE:  1 DECEMBER 1976      *
0015*  *                                                             *
0016*  *  MODIFICATIONS:                                             *
0017*  *    1:  ADD TYPC ROUTINE                                     *
0018*  *         7/14/77       W. BOWEN                              *
0019*  *    2:  ADD PNTL ROUTINE                                     *
0020*  *         7/28/77      W. BOWEN                               *
0021*  *    3:  CHANGE TYPC ROUTINE TO VAR. CHAR. COUNT
0022*  *        12/16/77     R. TASHJIAN
0023*  *                                                             *
0024*  *  MODULE FUNCTIONS:                                          *
0025*  *                                                             *
0026*  *    TYPI - PERFORMS KEYBOARD INPUT OF INTEGER STRINGS.       *
0027*  *                                                             *
0028*  *             CALL TYPI (I,IFLG)                              *
0029*  *                                                             *
```

```
*                I: LOCATION WHERE ENTERED VALUE IS STORED.               *
*             IFLG: LOCATION WHERE ERROR FLAG IS STORED.                  *
*                                                                         *
*      TYPA - PERFORMS KEYBOARD  INPUT OF ASCII CHARACTERS                *
*                                                                         *
*               CALL TYPA (AS,IFLGA)                                      *
*                                                                         *
*             AS: LOCATION WHERE ASCII CHARACTERS ARE STORED.             *
*             IFLGA: LOCATION WHERE ERROR FLAG IS STORED.                 *
*                                                                         *
*                IFLGA = NEGATIVE  - ERROR                                *
*                IFLGA = ZERO      - NO INPUT                             *
*                IFLGA = POSITIVE  - GOOD INPUT                           *
*                                                                         *
*      TYPC - PERFORMS KEYBOARD  INPUT OF UP TO TEN (10) ASCII            *
*               CHARACTERS.                                               *
*                                                                         *
*               CALL TYPC (AC,IFLGC,COUNT)                                *
*                                                                         *
*               AC: LOCATION OF BUFFER WHERE THE ASCII                    *
*                   CHARACTERS ARE STORED.                                *
*             IFLGC: LOCATION WHERE ERROR FLAG IS STORED.                 *
*                (IFLGC VALUES SAME AS TYPA'S IFLGA VALUES)               *
*             COUNT: NUMBER OF CHARACTERS ALLOWED                         *
*                                                                         *
*      PNTL - PRINTS ONE LINE OF  FORMATTED DATA WITHOUT DOING            *
*               A  CARRIAGE RETURN AND LINE FEED.                         *
*                                                                         *
```

```
*              CALL PNTL (IPNTB, IBLG)                                    *
*                                                                         *
*           IPNTB: PRINT BUFFER ADDRESS                                   *
*           IBLG:  BUFFER LENGTH                                          *
*                                                                         *
***************************************************************************


     *******************
     *   TYPI ROUTINE  *
     *******************

00000 000000  I     NOP
00001 000000  IFLG  NOP
00002 000000  TYPI  NOP
00003 016001X       JSB .ENTR
00004 000000R       DEF I
00005 016004X       JSB ENDIO     WAIT UNTIL OTHER I/O DONE
00006 000007R       DEF *+1
00007 103100        CLF 0
00010 062467R       LDA =D20      ALLOW 20 CHARACTERS
00011 066124R       LDB IBUF
00012 016251R       JSB READ
00013 002003        SZA,RSS
00014 026066R       JMP NOIN      NOTHING ENTERED
00015 052467R       CPA =D20
00016 026064R       JMP ERROR     20 OR MORE ENTERED, ERROR
00017 003004        CMA,INA
00020 072445R       STA CNTR
00021 066124R       LDB IBUF
00022 004066        CLE,ELB
00023 076444R       STB PNTR
```

```
0090*
0091   00024 016420R CHKI    JSB GETC       CHECK INPUT FOR LEGAL CHARS
0092   00025 052470R         CPA =B53       + ?
0093   00026 026045R         JMP OK
0094   00027 052471R         CPA =B55       - ?
0095   00030 026045R         JMP OK
0096   00031 052472R         CPA =B56        . ?
0097   00032 026045R         JMP OK
0098   00033 052473R         CPA =B57        / ?
0099   00034 026045R         JMP OK
0100   00035 052474R         CPA =B72        = ?
0101   00036 026045R         JMP OK
0102   00037 042475R         ADA =D-48      CHECK FOR RANGE 0-9
0103   00040 002020          SSA
0104   00041 026064R         JMP ERROR      <0, ERROR
0105   00042 042476R         ADA =D-10
0106   00043 002021          SSA,RSS
0107   00044 026064R         JMP ERROR      >9, ERROR
0108*
0109   00045 036444R OK      ISZ PNTR
0110   00046 036445R         ISZ CNTR       ALL CHARS CHECKED?
0111   00047 026024R         JMP CHKI       NO, LOOP
0112   00050 002400          CLA
0113   00051 006404          CLB,INB
```



```
0114   00052 016002X         JSB .DIO.
```

```
00053 000125R        DEF  IBUF+1
00054 000000         ABS  0
00055 000064R        DEF  ERROR
00056 016003X        JSB  .IOI.
00057 172000R        STA  I,I
00060 006404         CLB,INB          SET IFLG = +1
*
00061 176001R TYPIX  STB  IFLG,I
00062 102100         STF  0
00063 126002R        JMP  TYPI,I
*
00064 007400  ERROR  CCB              SET IFLG = -1
00065 026061R        JMP  TYPIX
*
00066 006400  NOIN   CLB              SET IFLG = 0
00067 026061R        JMP  TYPIX
*
*      ******************
*      *   TYPA ROUTINE    *
*      ******************
*
00070 000000  AS     NOP
00071 000000  IFLGA  NOP
00072 000000  TYPA   NOP
00073 016001X        JSB  .ENTR
00074 000070R        DEF  AS
00075 016004X        JSB  ENDIO       WAIT UNTIL OTHER I/O DONE
00076 000077R        DEF  *+1
00077 103100         CLF  0
00100 062477R        LDA  =D2
00101 066124R        LDB  IBUF
00102 016251R        JSB  READ
00103 002003         SZA,RSS
00104 026122R        JMP  NOINA       NOTHING ENTERED
00105 052500R        CPA  =D1
```

```
00106 002001         RSS
00107 026120R        JMP ERRA     ERROR, MORE THAN 1 CHAR ENTERED
00110 062125R        LDA IBUF+1
00111 001727         ALF,ALF      PUT CHARACTER IN RIGHT SIDE
00112 012501R        AND =B177
00113 172070R        STA AS,I
00114 006404         CLB,INB      SET IFLG = +1
*
00115 176071R TYPAX  STB IFLGA,I
00116 102100         STF 0
00117 126072R        JMP TYPA,I
*
00120 007400  ERRA   CCB          SET IFLG = -1
00121 026115R        JMP TYPAX
*
00122 006400  NOINA  CLB          SET IFLG = 0
00123 026115R        JMP TYPAX
*
00124 000125R IBUF   DEF *+1
00125 000000         BSS 21
```



```
*
    *******************
    *   TYPC ROUTINE  *
    *******************
*
```

```
00152 000000  AC     NOP
00153 000000  IFLGC  NOP
00154 000000  COUNT  NOP
00155 000000  TYPC   NOP
00156 016001X        JSB .ENTR      FACILITATE ARGUMENT TRANSFER
00157 000152R        DEF AC
00160 016004X        JSB ENDIO      WAIT UNTIL OTHER I/O IS DONE
00161 000162R        DEF *+1
00162 103100         CLF 0          CLEAR I/O FLAG
*
00163 162154R        LDA COUNT,I    ALLOW N ALPHANUMERIC CHARACTERS
00164 066152R        LDB AC
00165 016251R        JSB READ       READ INPUT DATA
00166 002003         SZA,RSS
00167 026203R        JMP NOINC      NOTHING ENTERED
00170 166154R        LDB COUNT,I
00171 006004         INB 2
00172 050001         CPA B
00173 002001         RSS 2
00174 026201R        JMP ERRN       ERROR, MORE THAN 10 CHARACTERS ENTE
RED
00175 006404         CLB,INB        SET IFLG = +1
*
00176 176153R TYPCX  STB IFLGC,I
00177 102100         STF 0          SET I/O FLAG
00200 126155R        JMP TYPC,I     RETURN
*
00201 007400  ERRN   CCB            SET IFLG = -1
00202 026176R        JMP TYPCX
*
00203 006400  NOINC  CLB            SET IFLG = 0
00204 026176R        JMP TYPCX
*
*      ******************
*      *   PNTL ROUTINE    *
```

```
0209*      ********************
0210*
0211   00205 000000  IPNTB NOP
0212   00206 000000  IBLG  NOP
0213   00207 000000  PNTL  NOP
0214   00210 016001X       JSB .ENTR        FACILITATE ARGUMENT TRANSFER
0215   00211 000205R       DEF IPNTB
0216   00212 016004X       JSB ENDIO        WAIT UNTIL OTHER I/O IS DONE
0217   00213 000214R       DEF *+1
0218   00214 103100        CLF 0            CLEAR I/O FLAG
0219*
0220   00215 066205R       LDB IPNTB        LOAD PRINT BUFFER ADDRESS POINTER

0221   00216 162206R       LDA IBLG,I       LOAD LENGTH OF PRINT BUFFER
0222*
0223   00217 004066        CLE,ELB          CHANGE TO CHARACTER POINTER
0224   00220 076444R       STB PNTR         SET-UP CHARACTER POINTER
0225   00221 002021        SSA,RSS          IS BUFFER LENGTH POSITIVE ?
```

PAGE 0006 #01  SUBROUTINES TYPI, TYPA, TYPC, AND PNTL

```
0226   00222 026225R       JMP CHAR         YES, CHARACTERS
0227   00223 003004        CMA,INA          NEGATIVE WORDS, MARE POSITIVE
0228   00224 001200        RAL              X2 (MAKE CHARACTERS)
0229*
0230   00225 003004  CHAR  CMA,INA
0231   00226 072445R       STA CNTR         SAVE COMPLEMENTED CHARACTER COUNT
```

```
      00227 062454R       LDA B12K
      00230 016430R       JSB XOTA       CONFIGURE I/O CARD
      00231 062445R       LDA CNTR
      00232 002003        SZA,RSS        COUNT INITIALLY ZERO ?
      00233 126207R       JMP PNTL,I     YES, RETURN
*
      00234 066444R NEXTP LDB PNTR
      00235 004065        CLE,ERB
      00236 160001        LDA B,I
      00237 002041        SEZ,RSS
      00240 001727        ALF,ALF
      00241 012456R       AND B177
*
      00242 016430R       JSB XOTA      SENT A CHARACTER
      00243 016433R       JSB XSFS      SET CONTROL AND WAIT FOR FLAG
      00244 036444R       ISZ PNTR      INCREMENT CHARACTER POINTER
      00245 036445R       ISZ CNTR      INCREMENT CHARACTER COUNT, SKIP =
0
      00246 026234R       JMP NEXTP
      00247 106711        CLC SC        CLEAR INTRRUPTS
      00250 126207R       JMP PNTL,I    OUTPUT COMPLETE, RETURN
*
     ****************************
     *   INTERNAL SUBROUTINES   *
     ****************************
*
      00251 000000  READ  NOP
      00252 004066        CLE,ELB      CHANGE TO CHAR POINTER
      00253 076440R       STB IPTR
      00254 076442R       STB SPTR
      00255 002021        SSA,RSS      POS?
      00256 026261R       JMP ICHAR    YES, CHARACTERS
      00257 003004        CMA,INA      NEG WORDS, MAKE POS
      00260 001200        RAL          X2
*
```

```
00261 003004  ICHAR CMA,INA
00262 072441R       STA ICTR
00263 072443R       STA SCTR
*
00264 062453R CONFI LDA B16K      CONFIGURE INTERFACE
00265 016430R       JSB XOTA
*
00266 016433R NEXTI JSB XSFS
00267 102511        LIA SC        INPUT
00270 012456R       AND B177
00271 052461R       CPA CR        CHECK FOR SPECIAL CHARACTERS
00272 026311R       JMP CARET
00273 052462R       CPA LF
00274 026266R       JMP NEXTI     IGNORE LF
00275 052463R       CPA BKSP
00276 026335R       JMP BACK
```

PAGE 0007 #01  SUBROUTINES TYPI, TYPA, TYPC, AND PNTL

```
00277 052464R       CPA DEL
00300 026325R       JMP DELET
00301 066441R       LDB ICTR
00302 006003        SZB,RSS       ICTR=0?
00303 026266R       JMP NEXTI     YES, DO NOT STORE
00304 016347R       JSB PUTC
00305 036440R       ISZ IPTR      INCREMENT BUFFER POINTER
00306 036441R       ISZ ICTR      INCREMENT CHARACTER COUNT
00307 000000        NOP
```

```
00310 026266R        JMP NEXTI      GO GET NEXT CHARACTER
*
00311 062454R CARET  LDA B12K       CONFIGURE FOR OUTPUT
00312 016430R        JSB XOTA
00313 062462R        LDA LF
00314 016430R        JSB XOTA       SEND LF
00315 016433R        JSB XSFS
00316 106711         CLC SC
00317 062460R        LDA SP         STORE SPACE TO MARK END OF INPUT
00320 016347R        JSB PUTC
*
00321 062443R IEXIT  LDA SCTR       # CHARACTERS TO A
00322 003004         CMA,INA
00323 042441R        ADA ICTR
00324 126251R        JMP READ,I     RETURN
*
00325 066465R DELET  LDB SLADD      PRINT REV SLASH, CR, LF
00326 002404         CLA,INA
00327 016364R        JSB PRINT
00330 062442R        LDA SPTR       RESTORE INITIAL POINTER, COUNTER
00331 072440R        STA IPTR
00332 062443R        LDA SCTR
00333 072441R        STA ICTR
00334 026264R        JMP CONFI
*
00335 062441R BACK   LDA ICTR
00336 052443R        CPA SCTR       COUNTER AT ORIGINAL VALUE?
00337 026266R        JMP NEXTI      YES, DO NOT MODIFY
00340 003400         CCA            BACK UP COUNTER, POINTER
00341 042441R        ADA ICTR
00342 072441R        STA ICTR       NO
00343 003400         CCA
00344 042440R        ADA IPTR
00345 072440R        STA IPTR
00346 026266R        JMP NEXTI
```

```
0326*
0327  00347 000000  PUTC  NOP           STORE CHARACTER
0328  00350 072364R       STA PRINT
0329  00351 066440R       LDB IPTR
0330  00352 004065        CLE,ERB
0331  00353 160001        LDA B,I
0332  00354 002041        SEZ,RSS
0333  00355 001727        ALF,ALF
0334  00356 012455R       AND B1774
0335  00357 032364R       IOR PRINT
0336  00360 002041        SEZ,RSS
0337  00361 001727        ALF,ALF
```

PAGE 0008 #01  SUBROUTINES TYPI, TYPA, TYPC, AND PNTL

```
0338  00362 170001        STA B,I
0339  00363 126347R       JMP PUTC,I
0340*
0341  00364 000000  PRINT NOP
0342  00365 004066        CLE,ELB       CHANGE TO CHAR POINTER
0343  00366 076444R       STB PNTR
0344  00367 002021        SSA,RSS       POS?
0345  00370 026373R       JMP CHARS     YES, CHARACTERS
0346  00371 003004        CMA,INA       NEG WORDS, MAKE POSITIVE
0347  00372 001200        RAL           X2
0348*
0349  00373 003004  CHARS CMA,INA
0350  00374 072445R       STA CNTR
```

```
00375 062454R        LDA B12K       CONFIGURE I/O CARD
00376 016430R        JSB XOTA
00377 062445R        LDA CNTR
00400 002003         SZA,RSS        COUNT INITIALLY ZERO?
00401 026410R        JMP NULL       YES, DO CR AND LF ONLY
*
00402 016420R NEXTO  JSB GETC
00403 016430R        JSB XOTA
00404 016433R        JSB XSFS
00405 036444R        ISZ PNTR
00406 036445R        ISZ CNTR
00407 026402R        JMP NEXTO
*
00410 062461R NULL   LDA CR         DONE, DO CR
00411 016430R        JSB XOTA
00412 016433R        JSB XSFS
00413 062462R        LDA LF         DO LF
00414 016430R        JSB XOTA
00415 016433R        JSB XSFS
00416 106711         CLC SC
00417 126364R        JMP PRINT,I    RETURN
*
00420 000000  GETC   NOP            GET CHARACTER FROM PACKED BUFFER
00421 066444R        LDB PNTR
00422 004065         CLE,ERB
00423 160001         LDA B,I
00424 002041         SEZ,RSS
00425 001727         ALF,ALF
00426 012456R        AND B177
00427 126420R        JMP GETC,I
*
00430 000000  XOTA   NOP            EXECUTE OTA
00431 102611         OTA SC
00432 126430R        JMP XOTA,I
*
```

```
00433 000000  XSFS   NOP            SET CONTROL AND WAIT FOR FLAG
00434 103711         STC SC,C
00435 102311         SFS SC
00436 026435R        JMP *-1
00437 126433R        JMP XSFS,I
*
*      ******************
*      *    LOCAL DATA    *
```

PAGE 0009 #01  SUBROUTINES TYPI, TYPA, TYPC, AND PNTL

```
*      ******************
*
00440 000000  IPTR   NOP
00441 000000  ICTR   NOP
00442 000000  SPTR   NOP
00443 000000  SCTR   NOP
00444 000000  PNTR   NOP
00445 000000  CNTR   NOP
00446 102300  SFS    SFS 0
00447 000100  B100   OCT 100
00450 000200  B200   OCT 200        BUFFER POINTER
00451 001100  B1100  OCT 1100       CHARACTER COUNTER
00452 003000  B3000  OCT 3000       START BUFFER POINTER
00453 160000  B16K   OCT 160000     TOTAL CHARACTER (AT START)
00454 120000  B12K   OCT 120000
00455 177400  B1774  OCT 177400
00456 000177  B177   OCT 177
```

```
0411  00457 000077  B77    OCT 77
0412  00460 000040  SP     OCT 40
0413  00461 000015  CR     OCT 15
0414  00462 000012  LF     OCT 12
0415  00463 000010  BKSP   OCT 10
0416  00464 000177  DEL    OCT 177
0417  00465 000466R SLADD  DEF RSL
0418  00466 056000  RSL    OCT 56000
0419  00011         SC     EQU 11B
0420  00000         A      EQU 0
0421  00001         B      EQU 1
0422*
      00467 000024
      00470 000053
      00471 000055
      00472 000056
      00473 000057
      00474 000072
      00475 177720
      00476 177766
      00477 000002
      00500 000001
      00501 000177
0423                       END
**  NO ERRORS *TOTAL **RTE ASMB 750420**
```

TYPI1S T=00004 IS ON CR00001 USING 00049 BLKS R=0371

```
ASMB,R,L
      HED SUBROUTINES TYPI, TYPA, TYPC, AND PNTL
      NAM TYPI1
      ENT TYPI,TYPA,TYPC,PNTL
      EXT .ENTR,.DIO.,.IOI.,END IO
*
*  ***********************************************************
*  *                                                          *
*  *  TITLE:    KEYBOARD INPUT AND SPECIAL LINE PRINTING      *
*  *                                                          *
*  *  MODULE:   TYPI        SOURCE:  TYPI1      VERSION:  X00 *
*  *                        OBJECT:  #TYPI1                   *
*  *                                                          *
*  *  AUTHOR:   B. TASHJIAN          DATE:  1 DECEMBER 1976   *
*  *                                                          *
*  *  MODIFICATIONS:                                          *
*  *    1:  ADD TYPC ROUTINE                                  *
*  *          7/14/77       W. BOWEN                          *
*  *    2:  ADD PNTL ROUTINE                                  *
*  *          7/28/77      W. BOWEN                           *
*  *                                                          *
*  *  MODULE FUNCTIONS:                                       *
*  *                                                          *
*  *    TYPI - PERFORMS KEYBOARD INPUT OF INTEGER STRINGS.    *
*  *                                                          *
*  *              CALL TYPI (I,IFLG)                          *
*  *                                                          *
*  *             I: LOCATION WHERE ENTERED VALUE IS STORED.   *
*  *          IFLG: LOCATION WHERE ERROR FLAG IS STORED.      *
*  *                                                          *
*  *    TYPA - PERFORMS KEYBOARD INPUT OF ASCII CHARACTERS    *
*  *                                                          *
```

```
*   *              CALL  TYPA (AS,IFLGA)                                      *
*   *                                                                         *
*   *          AS:  LOCATION  WHERE  ASCII  CHARACTERS  ARE  STORED.          *
*   *          IFLGA:  LOCATION  WHERE  ERROR  FLAG  IS  STORED.              *
*   *                                                                         *
*   *             IFLGA = NEGATIVE - ERROR                                    *
*   *             IFLGA = ZERO     - NO INPUT                                 *
*   *             IFLGA = POSITIVE - GOOD INPUT                               *
*   *                                                                         *
*   *      TYPC - PERFORMS KEYBOARD INPUT OF UP TO TEN (10) ASCII             *
*   *             CHARACTERS.                                                 *
*   *                                                                         *
*   *              CALL  TYPC (AC,IFLGC)                                      *
*   *                                                                         *
*   *              AC:  LOCATION  OF  BUFFER  WHERE  THE  ASCII               *
*   *                   CHARACTERS  ARE  STORED.                              *
*   *          IFLGC:  LOCATION  WHERE  ERROR  FLAG  IS  STORED.              *
*   *             (IFLGC  VALUES  SAME  AS  TYPA'S  IFLGA  VALUES)            *
*   *                                                                         *
*   *      PNTL - PRINTS ONE LINE OF FORMATTED DATA WITHOUT DOING             *
*   *             A  CARRIAGE  RETURN  AND  LINE  FEED.                       *
*   *                                                                         *
*   *              CALL  PNTL (IPNTB,IBLG)                                    *
*   *                                                                         *
*   *          IPNTB:  PRINT  BUFFER  ADDRESS                                 *
*   *          IBLG:   BUFFER  LENGTH                                         *
*   *                                                                         *
*   ***************************************************************************
*
*
*       *******************
*       *   TYPI ROUTINE  *
*       *******************
*
I       NOP
```

```
IFLG  NOP
TYPI  NOP
      JSB .ENTR
      DEF I
      JSB ENDIO     WAIT UNTIL OTHER I/O DONE
      DEF *+1
      CLF 0
      LDA =D20      ALLOW 20 CHARACTERS
      LDB IBUF
      JSB READ
      SZA,RSS
      JMP NOIN      NOTHING ENTERED
      CPA =D20
      JMP ERROR     20 OR MORE ENTERED, ERROR
      CMA,INA
      STA CNTR
      LDB IBUF
      CLE,ELB
      STB PNTR
*
CHKI  JSB GETC      CHECK INPUT FOR LEGAL CHARS
      CPA =B53      + ?
      JMP OK
      CPA =B55      - ?
      JMP OK
      CPA =B56       . ?
      JMP OK
      CPA =B57       / ?
      JMP OK
      CPA =B72       : ?
      JMP OK
      ADA =D-48     CHECK FOR RANGE 0-9
      SSA
      JMP ERROR     <0, ERROR
      ADA =D-10
```

```
         SSA,RSS
         JMP ERROR     >9, ERROR
*
OK       ISZ PNTR
         ISZ CNTR      ALL CHARS CHECKED?
         JMP CHKI      NO, LOOP
         CLA
         CLB,INB
         JSB .DIO.
         DEF IBUF+1
         ABS 0
         DEF ERROR
         JSB .IOI.
         STA I,I
         CLB,INB       SET IFLG = +1
*
TYPIX    STB IFLG,I
         STF 0
         JMP TYPI,I
*
ERROR    CCB           SET IFLG = -1
         JMP TYPIX
*
NOIN     CLB           SET IFLG = 0
         JMP TYPIX
*
*        ********************
*        *   TYPA ROUTINE   *
*        ********************
*
AS       NOP
IFLGA    NOP
TYPA     NOP
         JSB .ENTR
         DEF AS
```

```
      JSB ENDIO      WAIT UNTIL OTHER I/O DONE
      DEF *+1
      CLF 0
      LDA =D2
      LDB IBUF
      JSB READ
      SZA,RSS
      JMP NOINA      NOTHING ENTERED
      CPA =D1
      RSS
      JMP ERRA       ERROR, MORE  THAN 1 CHAR ENTERED
      LDA IBUF+1
      ALF,ALF        PUT CHARACTER IN RIGHT SIDE
      AND =B177
      STA AS,I
      CLB,INB        SET IFLG = +1
*
TYPAX STB IFLGA,I
      STF 0
      JMP TYRA,I
*
ERRA  CCB            SET IFLG = -1
      JMP TYRAX
*
NOINA CLB            SET IFLG = 0
      JMP TYRAX
*
IBUF  DEF *+1
      BSS 21
*
*     ********************
*     *   TYRC ROUTINE   *
*     ********************
*
AC    NOP
```

```
IFLGC NOP
TYPC  NOP
      JSB .ENTR    FACILITATE ARGUMENT TRANSFER
      DEF AC
      JSB ENDIO    WAIT UNTIL OTHER I/O IS DONE
      DEF *+1
      CLF 0        CLEAR I/O FLAG
*
      LDA =D10     ALLOW 10 ALPHANUMERIC CHARACTERS
      LDB AC
      JSB READ     READ INPUT DATA
      SZA,RSS
      JMP NOINC    NOTHING ENTERED
      CPA =D11
      RSS
      JMP ERRN     ERROR, MORE THAN 10 CHARACTERS ENTERED
      CLB,INB      SET IFLG = +1
*
TYPCX STB IFLGC,I
      STF 0        SET I/O FLAG
      JMP TYPC,I   RETURN
*
ERRN  CCB          SET IFLG = -1
      JMP TYPCX
*
NOINC CLB          SET IFLG = 0
      JMP TYPCX
*
*     ********************
*     *   PNTL ROUTINE   *
*     ********************
*
IPNTB NOP
IBLG  NOP
PNTL  NOP
```

```
      JSB .ENTR      FACILITATE ARGUMENT TRANSFER
      DEF IPNTB
      JSB ENDIO      WAIT UNTIL OTHER I/O IS DONE
      DEF *+1
      CLF 0          CLEAR I/O FLAG
*
      LDB IPNTB      LOAD PRINT BUFFER ADDRESS POINTER
      LDA IBLG,I     LOAD LENGTH OF PRINT BUFFER
*
      CLE,ELB        CHANGE TO CHARACTER POINTER
      STB PNTR       SET-UP CHARACTER POINTER
      SSA,RSS        IS BUFFER LENGTH POSITIVE ?
      JMP CHAR       YES, CHARACTERS
      CMA,INA        NEGATIVE WORDS, MARE POSITIVE
      RAL            X2 (MAKE CHARACTERS)
*
CHAR  CMA,INA
      STA CNTR       SAVE COMPLEMENTED CHARACTER COUNT
      LDA B12K
      JSB XOTA       CONFIGURE I/O CARD
      LDA CNTR
      SZA,RSS        COUNT INITIALLY ZERO ?
      JMP PNTL,I     YES, RETURN
*
NEXTP LDB PNTR
      CLE,ERB
      LDA B,I
      SEZ,RSS
      ALF,ALF
      AND B177
*
      JSB XOTA      SENT A CHARACTER
      JSB XSFS      SET CONTROL AND WAIT FOR FLAG
      ISZ PNTR      INCREMENT CHARACTER POINTER
      ISZ CNTR      INCREMENT CHARACTER COUNT, SKIP = 0
```

```
      JMP NEXTP
      CLC SC       CLEAR INTRRUPTS
      JMP PNTL,I   OUTPUT COMPLETE, RETURN
*
*     ***************************
*     *   INTERNAL SUBROUTINES  *
*     ***************************
*
READ  NOP
      CLE,ELB     CHANGE TO CHAR POINTER
      STB IPTR
      STB SPTR
      SSA,RSS     POS?
      JMP ICHAR   YES, CHARACTERS
      CMA,INA     NEG WORDS, MAKE POS
      RAL         X2
*
ICHAR CMA,INA
      STA ICTR
      STA SCTR
*
CONFI LDA B16K    CONFIGURE INTERFACE
      JSB XOTA
*
NEXTI JSB XSFS
      LIA SC      INPUT
      AND B177
      CPA CR      CHECK FOR SPECIAL CHARACTERS
      JMP CARET
      CPA LF
      JMP NEXTI   IGNORE LF
      CPA BKSP
      JMP BACK
      CPA DEL
      JMP DELET
```

```
      LDB ICTR
      SZB,RSS        ICTR=0?
      JMP NEXTI      YES, DO NOT STORE
      JSB PUTC
      ISZ IPTR       INCREMENT BUFFER POINTER
      ISZ ICTR       INCREMENT CHARACTER COUNT
      NOP
      JMP NEXTI      GO GET NEXT CHARACTER
*
CARET LDA B12K       CONFIGURE FOR OUTPUT
      JSB XOTA
      LDA LF
      JSB XOTA       SEND LF
      JSB XSFS
      CLC SC
      LDA SP         STORE SPACE TO MARK END OF INPUT
      JSB PUTC
*
IEXIT LDA SCTR       # CHARACTERS TO A
      CMA,INA
      ADA ICTR
      JMP READ,I     RETURN
*
DELET LDB SLADD      PRINT REV SLASH, CR, LF
      CLA,INA
      JSB PRINT
      LDA SPTR       RESTORE INITIAL POINTER, COUNTER
      STA IPTR
      LDA SCTR
      STA ICTR
      JMP CONFI
*
BACK  LDA ICTR
      CPA SCTR       COUNTER AT ORIGINAL VALUE?
      JMP NEXTI      YES, DO NOT MODIFY
```

```
        CCA              BACK UP COUNTER, POINTER
        ADA ICTR
        STA ICTR         NO
        CCA
        ADA IPTR
        STA IPTR
        JMP NEXTI
*
PUTC    NOP              STORE CHARACTER
        STA PRINT
        LDB IPTR
        CLE,ERB
        LDA B,I
        SEZ,RSS
        ALF,ALF
        AND B1774
        IOR PRINT
        SEZ,RSS
        ALF,ALF
        STA B,I
        JMP PUTC,I
*
PRINT   NOP
        CLE,ELB          CHANGE TO CHAR POINTER
        STB PNTR
        SSA,RSS          POS?
        JMP CHARS        YES, CHARACTERS
        CMA,INA          NEG WORDS, MAKE POSITIVE
        RAL              X2
*
CHARS   CMA,INA
        STA CNTR
        LDA B12K         CONFIGURE I/O CARD
        JSB XOTA
        LDA CNTR
```

```
      SZA,RSS      COUNT INITIALLY ZERO?
      JMP NULL     YES, DO CR AND LF ONLY
*
NEXTO JSB GETC
      JSB XOTA
      JSB XSFS
      ISZ PNTR
      ISZ CNTR
      JMP NEXTO
*
NULL  LDA CR       DONE, DO CR
      JSB XOTA
      JSB XSFS
      LDA LF       DO LF
      JSB XOTA
      JSB XSFS
      CLC SC
      JMP PRINT,I  RETURN
*
GETC  NOP          GET CHARACTER FROM PACKED BUFFER
      LDB PNTR
      CLE,ERB
      LDA B,I
      SEZ,RSS
      ALF,ALF
      AND B177
      JMP GETC,I
*
XOTA  NOP          EXECUTE OTA
      OTA SC
      JMP XOTA,I
*
XSFS  NOP          SET CONTROL AND WAIT FOR FLAG
      STC SC,C
      SFS SC
```

```
        JMP *-1
        JMP XSFS,I
*
*       *****************
*       *   LOCAL DATA  *
*       *****************
*
IPTR    NOP
ICTR    NOP
SPTR    NOP
SCTR    NOP
PNTR    NOP
CNTR    NOP
SFS     SFS 0
B100    OCT 100
B200    OCT 200          BUFFER POINTER
B1100   OCT 1100         CHARACTER COUNTER
B3000   OCT 3000         START BUFFER POINTER
B16K    OCT 160000       TOTAL CHARACTER (AT START)
B12K    OCT 120000
B1774   OCT 177400
B177    OCT 177
B77     OCT 77
SP      OCT 40
CR      OCT 15
LF      OCT 12
BKSP    OCT 10
DEL     OCT 177
SLADD   DEF RSL
RSL     OCT 56000
SC      EQU 11B
A       EQU 0
B       EQU 1
*
        END
```

USER PROGRAM STARTUP ROUTINE - USTRT

1. FUNCTION. USTRT provides the capability of starting a keyboard program immediately when a Fourier system is started without the need for operator action. The keyboard program usually starts a user program.

2. CONSTRAINTS. None.

3. CALLING SEQUENCE. Called by: CLOAD

```
        JMP     USTRT
```

4. DESCRIPTION OF INPUT. None.

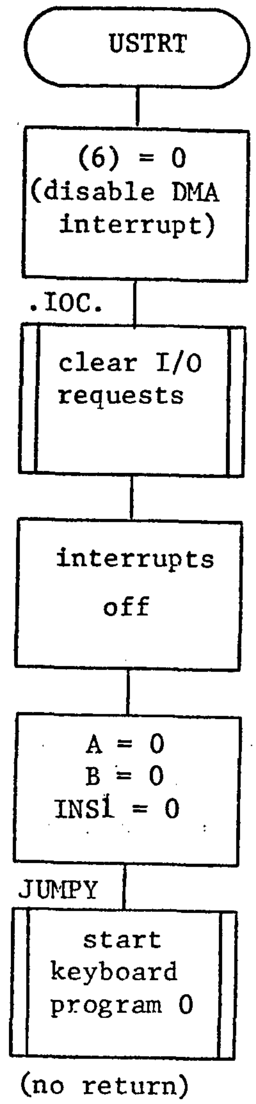
5. DESCRIPTION OF OUTPUT. None.

6. FILES USED. N/A

7. ERRORS AND RESTRICTIONS. None.

8. COMPUTER OPERATOR INSTRUCTIONS. N/A

9. DESCRIPTION OF PROCESSING. USTRT first sets address 6 to zero. This is necessary because USTRT is usually executed immediately after a core load, which leaves an interrupt request set in DMA 6. An intensive investigation disclosed no way of disabling this interrupt other than allowing it to occur and processing it. This is done by clearing address 6. Next, IOC is called to stop all I/O. Interrupts are disabled, A and B registers are cleared, location INS1 is set to zero and subroutine JUMPY is called. JUMPY starts the keyboard program whose label is in INS1, in this case, 0.


USTRT
(6) = 0
(disable DMA
interrupt)
.IOC.
clear I/O
requests
interrupts
off
A = 0
B = 0
INSi = 0
JUMPY
start
keyboard
program 0
(no return)

```
USTRT  T=00004 IS ON CR00001 USING 00002 BLKS R=0017

ASMB,R,L
       NAM USTRT
       ENT USTRT
       EXT .IOC.,INS1,JUMPY,START
USTRT  CLA
       STA 6       DISABLE DMA  INTERRUPT HALT
       JSB .IOC.
       OCT 0
       CLC 0
       CLB
       CLA
       STA INS1
       LDA JUMP    PUT JSB JUMPY AT END OF RUNS
       STA START
       JMP 2       EXECUTE RUNS  TO INIT WCS
JUMP   JSB JUMPY
       END

:
```

## 5.4 OLAAS USER PROGRAMS AND SUBROUTINES

The OLAAS System has various user programs and subroutines to accomplish certain users to aid in the analysis of the data. The following descriptions and listings are the programs and subroutines found in coreload 5:

Y7891 - Subroutine

Y7894 - User Program

Y7896 - User Program

Y7897 - User Program

Y7898 - Subroutine

Y7901 - User Program

Y7902 - User Program

Y7904 - User Program

Y7906 - User Program

Y7907 - User Program

Y0221 - User Program

BMDAT - Subroutine

Special User Program is not in coreload 5:

SOUND - User Program

Y7894 USER PROGRAM

This program fills up Fourier Data Blocks with beamformer data. The user may select beams 1-16. The data is loaded on a major frame basis using the subroutine BMDAT. The user may enter up to (8192/(BLOCK SIZE)) beams to be stored in blocks Ø to (8192/B.S.) - 1.

CALLING SEQUENCE: CALLED BY: Y7906 User Program

Call Y7894 (L, JBUF)

L = 1 (Read in scaled beam data (4Ø96 time domain words into block Ø)).

2 (Read in and sort data into blocks Ø-15, 256 words each)

3 (Read, sort, and printout cell JBUF)

JBUF = (Printout cell)

3/18/77

Y7S94S T=00004 IS ON CR00001 USING 00009 BLKS R=0090

```
FTN4,L
      SUBROUTINE Y7894(L,JBUF)
      DIMENSION JBUF(6),ICHAN(16),IDAT(16),NQUAL(5)
C
C       THIS ROUTINE FILLS UP FOURIER DATA BLOCKS WITH ACTIVE
C  BEAM DATA.
C
C      JBUF(1) AND JBUF(2) ARE THE BEAMFORMER CHANNELS READ.
C  (ONLY 1 CHANNEL NEED BE READ IF DESIRED)
C
C  DETERMINE BLOCK SIZE
C
      CALL GETQ(0,NQUAL)
      IBS=NQUAL(1)
C
C  DETERMINE THE NO. OF CHANNELS AND INITIALIZE THE ICHAN ARRAY,
C  WHICH IS THE ARRAY OF CHANNEL NUMBERS.
C
C
C  SET UP DISCRETE CHANNEL TRANSFER.
C
15    NC=L
      DO 22 I=1,NC
      ICHAN(I)=JBUF(I)
C
C  CHECK TO SEE IF CHANNEL .LE. 16
C
      IF(ICHAN(I).GE.17)GO TO 99
22    CONTINUE
C
C  CHECK FOR TOTAL DATA TRANSFER LESS THAN 8193 WORDS.
C
```

```
25    IF(IBS*NC.GE.8193)GO TO 99
C
C  SET UP CALIBRATION.
C  FIRST GET CURRENT VALUES.
C
      CALL GETQ(0,NQUAL)
C
C  SET SCALE FACTOR TO 10**1, COORDINATES TO LINEAR,REC., TIME.
C
      NQUAL(2)=100B
C
C  SET CALIBRATE WORD TO 16,383 TO CALIBRATE ERAPS DATA.
C
      NQUAL(3)=16383
C
C  SET FREQUENCY CODE TO 1 KHZ FMAX.
C
      NQUAL(4)=45
C
C  STORE AWAY IN ALL BLOCKS
C
      DO 60 I=1,NC
      IBLOK=I-1
60    CALL PUTQ(IBLOK,NQUAL)
C
C  SET UP FIRST 16 WORD ERAPS BEAM BLOCK TRANSFER..
C
      CALL BMDAT(IDAT,1)
C
C  READ IN AND DISTRIBUTE IBS ERAPS BEAM FRAMES..
C
      DO 40 ICHNL=0,(IBS-1)
C
C  GET 1 MINOR DATA FRAME
C
```

```
      CALL BMDAT(IDAT,2)
C
C  DISTRIBUTE.
C
      DO 30 IBLOK=0,(NC-1)
      IC=IBLOK+1
      IDATA=IDAT(ICHAN(IC))
30    CALL PUTW(IBLOK,ICHNL,IDATA,0)
40    CONTINUE
C
C  CLEAN UP.
C
      CALL BMDAT(IDAT,3)
      RETURN
C
C  ERROR RETURN.
C
99    WRITE(6,100)
100   FORMAT("PARAMETER ERROR IN CALL TO Y7894")
      CALL WHAT
      RETURN
      END
      END$
```

:

BMDATS T=00004 IS ON CR00001 USING 00008 BLKS R=0084

```
ASMB,R,L
      HED BMDAT
      NAM BMDAT
      ENT BMDAT
      EXT .ENTR
*
IDAT  NOP
KK    NOP
BMDAT NOP                ENTRY - FORTRAN CALLABLE
      JSB .ENTR
      DEF IDAT
      LDA KK,I
      CPA =D2
      JMP KK2
      CPA =D3
      JMP KK3
*
      CLF 0              KK=1, INTERRUPTS OFF
      LDA BUF1A          START INPUT
      JSB DMAI           INIT DMA
      STC SC,C
      STC 6,C            START DMA
      CLA,INA
      OTA SC             OUTPUT A 1 TO START
      JMP BMDAT,I        RETURN
*
KK2   SFS 6              KK=2, SWAP BUFFERS AND CONTINUE
      JMP *-1
      CLC SC
      CLC 6
      LDA BUF2A          SWAP BUFFERS
      LDB BUF1A
```

```
        STA BUF1A
        STB BUF2A
        JSB DMAI
        STC SC,C
        STC 6,C         START DMA
        LDB =D-16       MOVE LAST SET OF DATA
        LDA BUF2A
        STA PTR1
        LDA IDAT
        STA PTR2
MVNXT   LDA PTR1,I
        STA PTR2,I
        ISZ PTR1
        ISZ PTR2
        ISZ B
        JMP MVNXT       MOVE NEXT
        JMP BMDAT,I     RETURN
*
KK3     SFS 6           KK=3, WAIT FOR DMA, STOP
        JMP *-1
        CLC SC
        CLC 6
        CLA
        OTA SC
        STF 0           INTERRUPTS ON
        JMP BMDAT,I     RETURN
*
DMAI    NOP             INIT DMA SUBROUTINE
        STA B
        LDA SLCT2
        IOR =B120000
        OTA 6
        CLC 2
        LDA B
        IOR =B100000
```

```
      OTA 2
      STC 2
      LDA =D-16
      OTA 2
      JMP DMAI,I
*
PTR1  NOP
PTR2  NOP
BUF1A DEF BUF1
BUF2A DEF BUF2
BUF1  BSS 16
BUF2  BSS 16
SLCT2 DEF SC
SC    EQU 22B
A     EQU 0
B     EQU 1
      END
```

:

Y7896

This program fills up Fourier Analyzer data blocks with active ERAPS stave data. The user may select stave numbers 1-16, header word 1 (17), and header word 2 (18). The data is loaded on a major frame basis using the subroutine DMA1. The user may enter up to (8192/Block size) channels to be stored in blocks 0 to (8192/B.S.) - 1.

Calling sequence:

Y7896 K1 K2 K3

K1 = First Channel

K2 = Second Channel

Y7896S T=00004 IS ON CR00001 USING 00011 BLKS R=0010

```
FTN4,L
      SUBROUTINE Y7896 (NC,JBUF)
C
C  *************************************************************
C  *                                                           *
C  *  TITLE:  LOAD A FOURIER DATA BLOCK                        *
C  *                                                           *
C  *  MODULE: Y7896       SOURCE:  Y7896S      VERSION:  X00   *
C  *                                                           *
C  *  AUTHOR:  D. GOODMAN          DATE:  23 AUGUST 1976       *
C  *                                                           *
C  *                                                           *
C  *  MODULE FUNCTION:                                         *
C  *     THE Y7896 SUBROUTINE LOADS A FOURIER DATA BLOCK WITH  *
C  *  ACTIVE ERAPS DATA.   CHANNELS 1 -16 ARE DATA CHANNELS.   *
C  *  CHANNEL 17 IS THE FIRST HEADER WORD AND CHANNEL 18 IS    *
C  *  THE SECOND HEADER WORD.                                  *
C  *                                                           *
C  *  CALLING SEQUENCE:     CALLED BY - Y7906                  *
C  *                                                           *
C  *     CALL Y7896 (NC,JBUF)                                  *
C  *                                                           *
C  *  INPUT PARAMETERS:    NC   = NUMBER OF DATA BLOCK         *
C  *                       JBUF = INPUT BUFFER                 *
C  *                                                           *
C  *************************************************************
C
      DIMENSION JBUF(6),ICHAN(18),IDAT(72),NQUAL(5),ICHN(6)
C
C     DETERMINE BLOCK SIZE - STORE AS "IBS"
C
      CALL GETQ(0,NQUAL)
```

```
      IBS=NQUAL(1)
C
C     CHECK B.S. - CHANNEL NO. PRODUCT
C
      IF(IBS * NC .GE. 8193) GO TO 95
      IF (NC.EQ.0)GO TO 99
      IF (NC.GT.6)GO TO 99
C
C     REFORMAT KEYBOARD INPUT DATA
C
      DO 26 I = 1,NC
      IF(JBUF(I) .EQ. 17) GO TO 23
      IF(JBUF(I) .EQ. 18) GO TO 24
      ICHAN(I) = JBUF(I) + 2
      GO TO 25
   23 ICHAN(I) = 1
      GO TO 25
   24 ICHAN(I) = 2
   25 IF(ICHAN(I) .GT. 18) GO TO 99
   26 CONTINUE
C
C     SET UP CALIBRATION
C
C     SET SCALE FACTOR TO 10**1,COORDINATES TO LINEAR,REC,TIME
C
      NQUAL(2)=100B
C
C     SET CALIBRATOR WORD TO 16383 TO CALIBRATE ERAPS DATA
C
      NQUAL(3)=16383
C
C     SET FREQUENCY CODE TO 1 KHZ FMAX
C
      NQUAL(4)=45
C
```

```
C     STORE AWAY INTO ALL BLOCKS
C
      DO 60 I=1,NC
      IBLOK=I-1
   60 CALL PUTQ(IBLOK,NQUAL)
C
C     SET UP FIRST MAJOR FRAME TRANSFER
C
      CALL DMA1(IDAT,1)
C
C     READ IN AND DISTRIBUTE "IBS" MAJOR FRAMES
C
      DO 75 KL = 1,NC
   75 ICHN(KL) = 0
C
C     GET ONE MAJOR DATA FRAME
C
   70 CALL DMA1(IDAT,2)
C
C     DISTRIBUTE INTO BLOCKS
C
      DO 80 IC = 1,NC
      IBLOK=IC-1
      DO 80 JJ = 1,4
      INDEX =((JJ-1) * 18) + ICHAN(IC)
      IDATA = IDAT(INDEX)
      ICHNL = ICHN(IC)
      CALL PUTW(IBLOK,ICHNL,IDATA,0)
      ICHN(IC) = ICHN(IC) +1
      IF(ICHN(NC) .GE. IBS) GO TO 81
   80 CONTINUE
      IF(ICHNL .LT. IBS) GO TO 70
   81 CALL DMA1(IDAT,3)
C
C     TURN OFF DMA, TURN ON INTERRUPTS
```

```
C
      RETURN
C
C     ERROR RETURN
C
   95 WRITE(6,101)
  101 FORMAT(5X,"** ERROR, BLOCK SIZE TOO LARGE **")
      RETURN
   99 WRITE(6,100)
  100 FORMAT(5X,"** ERROR IN PARAMETER ENTRY **")
      RETURN
      END
      END$
```

Y7897 USER PROGRAM

This routine reads in 2Ø48 words of multiplexed beam data, sorts it into 16 blocks of 128 words each, Hanns and FFT's each block, normalizes each block to channel 1, and prints the log magnitude and phase of each channel relative to channel 1.

CALLING SEQUENCE:

Y7897 JBUF(1)

JBUF(1) = 1 - Read in scaled beam data (2Ø48 words to block 1)

2 - Reads in and sorts to block Ø-15 (128 words each)

3 - FFT's the data

SUBROUTINES CALLED:

Y7891 - Reads in scale Beam Data (2Ø48 words to block 1)

Y7898 - Sorts multiplexed Beam data.

Y7897S T=00004 IS ON CR00001 USING 00007 BLKS R=0054

```
FTN4,L
      SUBROUTINE Y7897(II,JBUF)
      DIMENSION JBUF(6),KBUF(1),NQUAL(5)
C
C     JBUF(1) = 1 TO READ IN SCALED BEAM DATA (2048 WDS TO BL 1)
C             = 2 TO READ IN AND SORT (TO BL 0-15, 128 WDS EA)
C             = 3 TO FFT THE DATA
C        THIS ROUTINE READS IN 2048 WORDS OF MULTIPLEXED BEAM DATA,
C  SORTS IT INTO 16 BLOCKS OF 128 WORDS,HANNS AND FFTS
C  EACH BLOCK, AND DISPLAYS THE RESULTS.
C
      IGO=JBUF(1)
C  BLOCKSIZE=2048 AND TIME DOMAIN
C
      NQUAL(1)=2048
      NQUAL(2)=100B
      NQUAL(3)=16383
      NQUAL(4)=45
      NQUAL(5)=0
      CALL PUTQ(0,NQUAL)
C
C  READ IN DATA
C
      I=1
      KBUF(1)=01
      CALL Y7891(I,KBUF)
C RETURN FOR READ IN ONLY
      IF(IGO     .EQ.1)RETURN
C
C  STORE 1
      CALL KYBD(054076B,1)
```

```
C   SORT
        CALL Y7898(1,KBUF)
        IF(IGO.EQ.2)RETURN
C   BLOCKSIZE =128
        NQUAL(1)=128
        DO 7 I=0,16
  7     CALL PUTQ(I,NQUAL)
C   RETURN FOR READ-IN AND SORT ONLY
C   HANN AND FFT EACH BLOCK
        DO 10 I=0,16
        CALL KYBD(044061B,I)
        CALL KYBD(043040B,I)
  10    CALL KYBD(052114B,I)
C   CHANGE BLOCK SIZE TO 2048
        CALL GETQ (0,NQUAL)
        NQUAL(1) = 2048
        CALL PUTQ (0,NQUAL)
        RETURN
        END
        END$
```

:

Y7891 USER SUBROUTINE

Reads in 2Ø48 words of multiplexed Beam Data into data block 1.

CALLING SEQUENCE:

Y7891 (K1, K2)

CALLED BY: Y7897 User Program

Y7891S T=00004 IS ON CR00001 USING 00004 BLKS R=0040

```
ASMB,R,L
      NAM Y7891
      ENT Y7891
      EXT .ENTR,DTAD0
NPAR  NOP
N1    NOP
Y7891 NOP
      JSB .ENTR
      DEF NPAR
      CLF 0          INT OFF
      CLA
      OTA SC2
      LDA SLCT2      GET SELECT CODE
      IOR =B120000
      OTA 6          OUTPUT CW1
      CLC 2
      LDA DTAD0
      IOR =B100000
      OTA 2          OUTPUT CW2
      STC 2
      LDA =D-2048
      OTA 2          OUTPUT CW3
      STC SC2,C      START SC2
      STC 6,C        START DMA
      CLA,INA
      OTA SC2        SET OUTPUT BIT 0 ON SC2
      SFS 6          WAIT FOR DMA
      JMP *-1
      CLC SC2
      CLC 6
      CLA
      OTA SC2        CLEAR OUTPUT BIT 0 ON SC2
```

```
              STF  0          INT ON
              JMP  Y7891,I    RETURN
       N      NOP
       CTR1   NOP
       CTR2   NOP
       SLCT2  DEF  SC2
       SC2    EQU  22B
              END
```

Y7898 USER PROGRAM SUBROUTINE

This subroutine is used with Y7897 to sort 2Ø48 multiplexed Beam words in block 1 and put sorted data into block Ø. The data is sorted so that consecutive channels occupy adjacent 128 word cells in the 2Ø48 data block.

CALLING SEQUENCE: CALLED BY: Y7897 User Program

Y7898 (K1)

K1 - Dummy variable, not used in subroutine.

Y789SS T=00004 IS ON CR00001 USING 00003 BLKS R=0019

```
FTN4,L
      SUBROUTINE Y7898(II,JBUF)
      DIMENSION JBUF(1)
C
C        THIS ROUTINE SORTS 2048 MULTIPLEXED BEAM WORDS
C  (IN BLOCK 1) AND PUTS SORTED DATA IN BLOCK 0.
C
C  K=WORD COUNTER FOR BLOCK 0
      K=0
      DO 20 I=0,15
      DO 20 J=0,127
      ICHNL=I+16*J
      CALL GETW(1,ICHNL,IDTAR,IDTAI)
      ICHNL=K
      CALL PUTW(0,ICHNL,IDTAR,IDTAI)
20    K=K+1
      RETURN
      END
      END$
```

:

## Y7901

This subroutine accesses disc-loaded Beamformer data as collected by the assembly language routine RDATA.

Calling sequence:

Y7901 K1 K2 K3

K1 = Ping number

K2 = Beam number

K3 = Index number (1 - 128 word; Index = 0.192 sec)

Y7901S"T=00004 IS ON CR00001 USING 00004 BLKS R=0027

10-4-76

```
FTN4,L
C
C     ***************************************
C     *                                     *
C     * THIS SUBROUTINE ACCESSES DISC-LOADED *
C     * BEAM-FORMER DATA AS COLLECTED BY THE *
C     * ROUTINE RDATA.                      *
C     *                                     *
C     * USER-ENTERED PARAMETERS-            *
C     *                                     *
C     *      KBUF(1) = PING NUMBER          *
C     *      KBUF(2) = BEAM NUMBER          *
C     *      KBUF(3) = INDEX NUMBER         *
C     *                                     *
C     ***************************************
C
      SUBROUTINE Y7901(KK,KBUF)
      DIMENSION KBUF(3),NQUAL(5)
      K1=KBUF(1)
      K2=KBUF(2)
      K3=KBUF(3)
C
C     SET SCALE FACTORS (10**1, LINEAR, RECT., TIME)
C
      CALL GETQ( 0, NQUAL)
      K4= NQUAL(1)
      NQUAL(2)=100B
      NQUAL(3)=16383
      NQUAL(4)=45
      NQUAL(5)=0
      CALL PUTQ(0,NQUAL)
C
```

10-4-76

```
C       READ IN DATA
C
         CALL RDATA(K1,K2,K3,K4,K5)
        IF(K5.EQ.0)RETURN
        WRITE(2,100)
  100   FORMAT(5X,"STATUS IS NON-ZERO, NO DATA RETURNED")
        RETURN
        END
        ENDS
```

:

## Y7902

The program prints out pertinent header data concerning Beamformer data recorded on the disc.

Calling sequence:

Y7902 K1

K1 = Ping number

3-18-7

```
Y7902S T=00004 IS ON CR00001 USING 00008 BLKS R=0058

FTN4,L
C
C     ************************************************************
C     *                                                          *
C     *  THIS PROGRAM READS IN HEADER DATA BASED ON              *
C     *  THE BEAMFORMER INFORMATION RECORDED ON THE              *
C     *  DISC. CALLING SEQUENCE : Y7902 IPNG                     *
C     *      (IPNG IS THE PARTICULAR PING NUMBER)                *
C     *                                                          *
C     ************************************************************
C
      SUBROUTINE Y7902(L,JBUF)
      INTEGER DATA
      DIMENSION JBUF(1),DATA(16),MASK(4),IDTA(6)
      MASK(1) = 17B
      MASK(2) = 360B
      MASK(3) = 7400B
      MASK(4) = 170000B
      CALL HEADR(1,DATA)
      IF(JBUF(1) .GT. DATA(10)) GO TO 700
      IPING=JBUF(1)
      CALL HEADR(IPING,DATA)
C
C     CHECK FOR LOCATION OF SOURCE
C
      IF(DATA(9) .EQ. 1) GO TO 70
      WRITE(2,103)
  103 FORMAT(5X,"SOURCE IS TAPE RECORDER")
      GO TO 80
   70 WRITE(2,104)
  104 FORMAT(5X,"SOURCE IS ON-LINE")
   80 CONTINUE
```

```
C
C     CALCULATE THE TIME
C
       NWORD = DATA(2)
      DO 20 I = 1,4
      MSK = MASK(I)
   20 IDTA(I) = IAND(MSK,NWORD)
      NWORD = DATA(1)
      DO 30 I = 1,2
      MSK = MASK(I)
      J = I + 4
   30 IDTA(J) = IAND(MSK,NWORD)
      KT1 = ((IDTA(6) / 20B) * 10) + IDTA(5)
      KT2 = ((IDTA(4) / 10000B) * 10) + (IDTA(3)/400B)
      KT3 = ((IDTA(2) / 20B) * 10) + IDTA(1)
C
C     WRITE THE HEADER
C
      WRITE(2,100)KT1,KT2,KT3,DATA(3)
  100 FORMAT(5X,"TIME = ",I2,":",I2,":",I2,5X,"COMPASS HEADING =",
     1I4," DEGREES")
      WRITE(2,101) DATA(5),DATA(6),DATA(8),DATA(7),DATA(10)
  101 FORMAT(5X,"FIRST BEAM =",I3,2X,"AT",I4," DEGREES",/,5X,
     1"BEAMS RECORDED =",I4,5X,"PING NUMBER",I4," OF",I4)
      WRITE(2,102) DATA(12),DATA(11)
  102 FORMAT(5X,"RECORDING TIME LENGTH",I3," SECONDS AFTER DELAY O
     1I3," SECONDS")
      RETURN
  700 WRITE(2,701)JBUF(1)
  701 FORMAT("PING",I3," WAS NOT RECORDED")
      RETURN
      END
      END$
```

## Y7904

This allows the user to print out the decimal value of each cell in a chosen block. The cells printed out are determined by answers the user gives to questions.

Calling sequence:

Y7904 (No additional parameters necessary)

Y7904S T=00004 IS ON CR00001 USING 00007 BLKS R=0046

3/24/77

```
FTN4,L
      SUBROUTINE Y7904(L,JBUF)
C
C     THIS ROUTINE IS USED TO EXAMINE THE CONTENTS OF A
C     FOURIER DATA BLOCK. USER-ENTERED PARAMETERS DIRECT THE
C     PROCESSING.
C
      DIMENSION JBUF(2),KOUT(8),NQUAL(5)
      DATA IFCN/1/, IBS/1024/,  IBLK/0/, IST/0/, KWRDS/150/
      WRITE(2,21)
   21 FORMAT("ENTER APPROPRIATE NUMBER",/,5X,"1 - WRITE EXISTING "
     1"FOURIER DATA BLOCK",/,5X,"2 - WRITE DISC-STORED DATA BLOCK"
      CALL INPT(IFCN,1,2)
      WRITE(2,22)
   22 FORMAT("ENTER BLOCK SIZE")
      CALL INPT(IBS,64,4096)
      CALL GETQ(0,NQUAL)
      NQUAL(1) = IBS
      CALL PUTQ(0,NQUAL)
      IF(IFCN .EQ. 2) GO TO 23
      WRITE(2,24)
   24 FORMAT("ENTER NUMBER OF BLOCK TO BE PRINTED OUT")
      KMAX = (8192/IBS) - 1
      CALL INPT(IBLK,0,KMAX)
      GO TO 29
   23 IBLK = 0
   29 WRITE(2,30)
   30 FORMAT("ENTER FIRST WORD TO BE PRINTED")
      CALL INPT(IST,0,IBS)
      WRITE(2,31)
   31 FORMAT("ENTER NUMBER OF CONSECUTIVE WORDS TO BE PRINTED")
      CALL INPT(KWRDS,1,IBS)
```

```
      IF(IFCN .EQ. 1) GO TO 25
      CALL KYBD(2HMS,31,0)
      CALL KYBD(2HMS,11)
   25 DO 26 K=IST,(KWRDS+IST-1),8
      DO 27 ICHNL=K,(K+8)
      INDEX=ICHNL-K+1
      CALL GETW(0,ICHNL,IDTA,INOP)
   27 KOUT(INDEX)=IDTA
      WRITE(2,28)K,(KOUT(I),I=1,8)
   28 FORMAT(    "(",I4,")",8(1X,I6))
   26 CONTINUE
   32 RETURN
      END
      END$
```

## Y7906

This program allows the operator to select stave data, beam data, or analog input for analysis. This data is Fourier transformed, then averaged and Hanned a preset number of times (entered by the user). The resulting averaged power spectrum is then displayed.

Calling sequence:

Y7906 (No additional parameters necessary)

8/24/77

Y7906S T=00004 IS ON CR00100 USING 00005 BLKS R=0018

```
FTN4
      SUBROUTINE Y7906(L,JBUF)
C
C     ***************************************************
C     *                                                 *
C     *  THIS ROUTINE ALLOWS THE OPERATOR TO ENTER      *
C     *  VARIOUS PARAMETERS IN ORDER TO REDUCE THE      *
C     *  NUMBER OF KEYBOARD ENTRIES NEEDED FOR AVERAGING *
C     *  DEU CHANNEL DATA.  THE USER ENTERED PARAMETERS  *
C     *  ARE--                                          *
C     *      STAVE,BEAM OR ANALOG INPUT                 *
C     *      CHANNEL NO. FOR STAVE AND BEAM ANALYSIS    *
C     *                                                 *
C     *  OTHER PARAMETERS ENTERED LATER ARE AS FOLLOWS:  *
C     *         NUMBER OF AVERAGES REQUIRED             *
C     *         BLOCK SIZE                              *
C     *         NUMBER OF HANNINGS NEEDED               *
C     *                                                 *
C     ***************************************************
C
      DIMENSION JBUF(1),NQUAL(5),LBUF(6),NBUF(6)
      DATA IBEL/000007B/,IAN/1/,IAV/10/,IHAN/1/,ISTV/5/,
     CIBEAM/13/,IBS/1024/
C
C     SET BLOCK SCALE FACTORS
C
      CALL GETQ(0,NQUAL)
      NQUAL(1) = 1024
      NQUAL(2) = 1003
      NQUAL(3) = 16383
      NQUAL(4) = 45
      CALL PUTQ(0,NQUAL)
```

```
C
C     DETERMINE WHETHER TO USE ANALOG INPUT, Y7896, OR Y7894
C
      WRITE(1,5)
    5 FORMAT("PLEASE ENTER THE TYPE OF ANALYSIS DESIRED: ",/,
     1"FOR STAVE INPUT, ENTER 1 ",/,"FOR BEAM INPUT, ENTER 2 ",
     2/,"FOR ANALOG INPUT, ENTER 3 ")
      CALL INPT(IAN,1,3)
      IF(IAN.EQ.3)GO TO 10
      IF(IAN.EQ.2)GO TO 8
    6 WRITE(1,7)
    7 FORMAT("ENTER THE STAVE NUMBER: ")
      CALL INPT(ISTV,1,16)
      LBUF(1)=ISTV
      GO TO 14
    8 WRITE(1,9)
    9 FORMAT("ENTER THE BEAM NUMBER: ")
      CALL INPT(IBEAM,1,16)
      NBUF(1)=IBEAM
      GO TO 14
   10 WRITE(1,11)
   11 FORMAT("PLEASE SET THE SAMPLE RATE AND THE INPUT LEVEL ",/,
     C"CONTROLS ON THE ANALOG TO DIGITAL CONVERTER. ")
   14 WRITE(1,15)
   15 FORMAT("PLEASE ENTER THE BLOCK SIZE DESIRED: ")
   16 N=6
      CALL INPT(IBS,64,4096)
   17 IF(N.EQ.13)GO TO 18
      IHH=(2**N)
      IF(IBS.EQ.IHH)GO TO 22
      N=N+1
      GO TO 17
   18 WRITE (1,19)
   19 FORMAT("YOU STUPID TURKEY!!!",/,"TRY AGAIN!!!")
      DO 20 JK=1,10,1
```

```
   20 WRITE(1,21)IBEL
   21 FORMAT(1A2)
      CONTINUE
      GO TO 16
   22 CALL GETQ(0,NQUAL)
      NQUAL(1) = IBS
      CALL PUTQ(0,NQUAL)
C
C     GET NUMBER OF AVERAGES AND NUMBER OF HANNINGS DESIRED
C
      WRITE(1,23)
   23 FORMAT("ENTER THE NUMBER OF TIMES THAT YOU WANT ME TO ",/,
     C"AVERAGE THE INPUT: ")
      CALL INPT(IAV,1,10000)
      WRITE(1,24)
   24 FORMAT("HOW MANY HANNINGS WITHIN EACH AVERAGING LOOP??")
      CALL INPT(IHAN,0,10000)
      WRITE(1,200)
  200 FORMAT("WORKING.......")
C
C     CLEAR OUT THE BLOCKS
C
      CALL KYBD(041514B,0)
      CALL KYBD(041514B,1)
C
C     START OF AVERAGING LOOP
C
      DO 45 I=1,IAV,1
      IF (IAN.EQ.3)GO TO 25
      IF (IAN.EQ.2)GO TO 245
      CALL Y7896(1,LBUF)
      GO TO 27
  245 CALL Y7894(1,NBUF)
      GO TO 27
C
```

```
C        CALL KEYBOARD ENTRIES
C
     25 CALL KYBD(051101B,0)
     27 IF(IHAN.EQ.0)GO TO 29
        DO 28 J=1,IHAN,1
     28 CALL KYBD(044061B,0)
        CONTINUE
     29 CALL KYBD(043040B,0)
        CALL KYBD(2H*-,0)
        CALL KYBD(2HA+,1)
C
C       STORE RUNNING TOTAL IN BLOCK ONE
C
        CALL KYBD(2HX>,1)
     45 CONTINUE
C
C       LOAD TOTAL INTO BLOCK ZERO
C
        CALL KYBD(2HX<,1)
C
C       DIVIDE TOTAL BY NUMBER OF AVERAGES
C
        CALL KYBD(2H: ,0,IAV)
C
C       TAKE LOG MAGNITUDE OF BLOCK ZERO
C
        CALL KYBD(052114B,0)
        WRITE(1,46)
     46 FORMAT("DONE!!!")
        WRITE(1,47)IBEL
     47 FORMAT(1A2)
C
C       DISPLAY BLOCK ZERO
C
        CALL KYBD(042040B,0)
```

```
0138          RETURN
0139       50 END
0140          END$
```

Y7907 USER PROGRAM

This subroutine tests channel response for all 16 DEU channels. However, all channel inputs must be tied to the same signal source. The user entered parameters are as followers:

F1 or F2 DEU

Test vessel pressure (PSI)

Data Channel block number (Use Y7896 or Y7906 along with Y∅221 to determine)

CALLING SEQUENCE:

Y79∅7 (stand alone routine)

SUBROUTINES CALLED:

Y7896 & INPT

Y7907S T=00004 IS ON CR00001 USING 00020 BLKS R=0115

```
FTN4,L
      SUBROUTINE Y7907(L,JBUF)
      DIMENSION JBUF(5),NQUAL(5)
      DATA DELT/8.333333333E-5/,IBEL/000007B/,IFR/2/,IFRE1/0000/,
     1IPRES/14/,IBLK/000/,IREF/0/,INUM/15/,NQUAL(1)/1024/,
     2IDA/103/,IDB/257/,IDC/411/,IDD/408/,IDE/254/,IDF/101/
C
C     ************************************************
C     *                                              *
C     *  THIS ROUTINE TESTS CHANNEL RESPONSE FOR     *
C     *  ALL 16 DEU CHANNELS. HOWEVER, ALL CHANNEL   *
C     *  INPUTS MUST BE TIED TO THE SAME SIGNAL      *
C     *  SOURCE. THE USER ENTERED PARAMETERS ARE:    *
C     *     F1 OR F2 DEU                             *
C     *     INPUT FREQUENCY                          *
C     *     TEST VESSEL PRESSURE (PSI)               *
C     *     DATA CHANNEL BLOCK NUMBER                *
C     *     (USE Y7896 OR Y7906 ALONG WITH Y0221     *
C     *                 TO DETERMINE)                *
C     *                                              *
C     ************************************************
C
      CALL PUTQ(0,NQUAL)
      WRITE(1,10)
   10 FORMAT("WHAT IS THE CENTER FREQUENCY FOR THE DEU UNDER TEST?
     1"FOR AN F1 DEU,ENTER 1",/,"FOR AN F2 DEU,ENTER 2")
      CALL INPT(IFR,1,2)
      IF(IFR.EQ.1)IFRE=((27960+40)/7)
      IF(IFR.EQ.2)IFRE=((1596+4)*3)
      WRITE(1,11)
   11 FORMAT("PLEASE ENTER THE SIGNAL FREQUENCY:")
      CALL INPT(IFRE1,3700,5100)
```

```
      WRITE(1,145)
  145 FORMAT("ENTER THE PRESSURE LEVEL FOR THIS TEST (PSI):")
      CALL INPT(IPRES,0,10000)
      IFRE3=IFRE1-IFRE
      IF(IFR.EQ.2)GO TO 147
      IFRE2=166+IFRE3
      GO TO 148
 147  IFRE2=166-IFRE3
 148  IF(IFR.EQ.2)GO TO 110
      IF(IFRE1.EQ.(IFRE-100))IBLK=IDA
      IF(IFRE1.EQ.IFRE)IBLK=IDB
      IF(IFRE1.EQ.(IFRE+100))IBLK=IDC
      GO TO 112
 110  IF(IFRE1.EQ.(IFRE-100))IBLK=IDD
      IF(IFRE1.EQ.IFRE)IBLK=IDE
      IF(IFRE1.EQ.(IFRE+100))IBLK=IDF
 112  IF(IFRE1.NE.(IFRE-100).AND.IFRE1.NE.IFRE.AND.IFRE1.NE.
     1(IFRE+100))GO TO 114
      GO TO 215
 114  WRITE(1,12)
   12 FORMAT("NOW ENTER THE CHANNEL BLOCK NUMBER FOR THIS ",/,
     1"FREQUENCY:")
      CALL INPT(IBLK,0,500)
  215 WRITE(1,115)
 115  FORMAT("DO YOU WISH TO SPECIFY THE REFERENCE CHANNEL ?",/,
     1"(1 OR 0)")
      CALL INPT(IREF,0,1)
      IF(IREF.EQ.1)GO TO 120
      INUM=1
      GO TO 125
 120  WRITE(1,121)
 121  FORMAT("ENTER THE REFERENCE CHANNEL NUMBER:")
      CALL INPT(INUM,1,16)
 125  DO 75 JK=1,16,1
      JBUF(1)=JK
```

```
      JBUF(2)=INUM
      CALL Y7896(2,JBUF)
      IF(JK.EQ.1)GO TO 16
      GO TO 18
   16 CALL CHOUT(33B)
      CALL CHOUT(14B)
      WRITE(1,17)
   17 FORMAT(///,25X,"DEU CHANNEL RESPONSE")
      IF(IFRE1.LT.IFRE)GO TO 175
      IF(IFRE1.EQ.IFRE)GO TO 177
      GO TO 179
  175 WRITE(1,176)IFR,IFRE3,IPRES
  176 FORMAT(//,10X,"FREQUENCY=F",I1,I4,10X,"PRESSURE=",I4," PSI")
      GO TO 200
  177 WRITE(1,178)IFR,IPRES
  178 FORMAT(//,10X,"FREQUENCY=F",I1,15X,"PRESSURE= ",I5," PSI")
      GO TO 200
  179 WRITE(1,180)IFR,IFRE3,IPRES
  180 FORMAT(//,10X,"FREQUENCY=F",I1,"+",I3,10X,"PRESSURE= ",I5,
     1" PSI")
  200 WRITE(1,201)
  201 FORMAT(2X,"************************************",
     1"***************************** ",//,5X,"CHANNEL",
     24X,"AMPLITUDE DIFFERENCE",2X,"PHASE DIFFERENCE",5X,
     3"COMMENTS",/)
   18 CALL KYBD(044061B,0)
      CALL KYBD(044061B,1)
      CALL KYBD(043040B,0,1)
      CALL KYBD(035040B,1)
      CALL KYBD(052120B)
      CALL GET(0,IBLK,DATAR,DATAI)
      CALL KYBD(052114B)
      DATA=20.*(ALOGT(DATAR))
   29 JKL=JK-INUM
   32 AJKL=JKL
```

```
      FREQ2=IFRE2
      SHFT=360.0*FREQ2*AJKL*DELT
      PHAS=DATAI-SHFT
      PHAS1=ABS(PHAS)
      IF (PHAS1.GT.10.)GO TO 60
      DATA1=ABS(DATA)
      IF (DATA1.GT.1.0)GO TO 60
      IF(JK.EQ.INUM)GO TO 48
      GO TO 50
   48 WRITE(1,49)JK,DATA,PHAS
   49 FORMAT(6X,I2,13X,F5.2," DB",11X,F6.2," DEGREES",5X,
     1"** REF **")
      GO TO 75
   50 WRITE(1,51)JK,DATA,PHAS
   51 FORMAT(6X,I2,13X,F5.2," DB",11X,F6.2," DEGREES")
      GO TO 75
   60 WRITE(1,61)JK,DATA,PHAS,IBEL
   61 FORMAT  (6X,I2,13X,F5.2," DB",11X,F6.2," DEGREES",5X,
     1"* CHECK *",A2)
   75 CONTINUE
      CALL TPLOT(0,25,125)
      CALL TPLOT(1,25,610)
      CALL TPLOT(1,1000,610)
      CALL TPLOT(1,1000,125)
      CALL TPLOT(1,25,125)
      CALL TPLOT(0,200,125)
      CALL TPLOT(1,200,610)
      CALL TPLOT(0,510,610)
      CALL TPLOT(1,510,125)
      CALL TPLOT(0,800,125)
      CALL TPLOT(1,800,610)
      CALL TPLOT(0,25,548)
      CALL TPLOT(1,1000,548)
   99 WRITE(1,100)IBEL
  100 FORMAT(A2)
```

```
END
END$

:
```

HEWLETT-PACKARD FOURIER ANALYZER USER'S GROUP

CONTRIBUTED PROGRAM DISCLAIMER

[X] INITIAL SUBMISSION [ ] REVISION (PROGRAM NO. )

CLASSIFICATION CODE (a list of classification codes is included on next page)

Y | 0 | 2 | 2 | 1 | "KROSS"

Classification

TITLE DATA BLOCK CURSOR ROUTINE

CONTRIBUTOR'S NAME DEAN JOHNSON

ORGANIZATION HEWLETT-PACKARD
SANTA CLARA DIVISION

ADDRESS 5301 STEVENS CREEK BLVD.
SANTA CLARA, CALIF. 95050

TELEPHONE (408) 246-4300 EXT-2566

Name and Organization to appear in catalog? [X] Yes [ ] No

May a user contact you directly? [X] Yes [ ] No

MATERIAL SUBMITTED

[X] Documentation [X] Source Program [ ] Test Case

[X] Other LIBRARY

ACKNOWLEDGEMENT AND AGREEMENT:

To the best of my knowledge, this contributed program is free of any proprietary information belonging to any person or organization. I am making this program information available to the HEWLETT-PACKARD Fourier Analyzer User's Group. I hereby agree that Hewlett-Packard may reproduce, publish, and use it, and authorize others to do so without obligations or liability of any kind.

(Signature) (Date)

HEWLETT-PACKARD FOURIER ANALYZER USER'S GROUP

CONTRIBUTED PROGRAM DOCUMENTATION FORM

1.0 IDENTIFICATION

10637-80221

1.1 Program Title

DATA BLOCK CURSOR ROUTINE

1.2 User Program Command Number

Y0221

1.3 Program Language

[ ] FORTRAN II

[X] Assembly Language

[ ] FORTRAN IV

2.0 PROGRAM DESCRIPTION

When called, Y0221 places a cursor across any frequency domain data block displayed ~~(time domain data is ignored)~~. The cursor can be moved left or right, and information - channel #, block #, frequency, and amplitude - can be printed out by pressing display register buttons. The program works on linear or log, rectangular or polar, and on partial displays. If Y0221 is called a second time, the program ends.

The display register controls are:

BIT 7 = FAST RIGHT MOVEMENT
" 8 = STEP RIGHT
" 9 = PRINT INFO
" 10 = HIGH SPEED DISPLAY W/OUT CURSOR
" 11 = ZOOM
" 12 = STEP LEFT
" 13 = FAST LEFT MOVEMENT

Time Data

CHANNEL nnn OF BLOCK nnnn

TIME = nnnn *10** ±nn SEC

AMPLITUDE = nnnn *10** ±nnn

Freq. Data

← Same

FREQUENCY = nnnn *10** ±nn HZ

← Same

CONTRIBUTED PROGRAM DOCUMENTATION FORM

## 3.0 ENVIRONMENT DESCRIPTION

### 3.1 Supported Software Requirements

[X] 5451 B Software Release ___

[ ] 5451__/71__Software Release ___

[ ] 5451__/70__Software Release ___

### 3.2 Non-Supported Software Requirements

Name and order number of other software packages called by the contributed program:

NONE

### 3.3 Storage Requirements

FORTRAN, Assembly Language (to be completed by User's Group)

$(2270)_{10}$ without external subprograms

$(442)_{10}$ with external subprograms

$(93)_{10}$ Basepage

### 3.4 Hardware Requirements

5451B FOURIER SYSTEM

CONTRIBUTED PROGRAM DOCUMENTATION FORM

## 4.0 USER PROCEDURES

### 4.1 Tape Identification

Account for all physical tapes included in this package.

| Tape Number | Contents<br>Programs, tests, data, etc., in order from outside to inside of tape | Control Statement (if any) |
|---|---|---|
| 1 | Y0221 SOURCE TAPE<br>KROSS<br>TTY1<br>HASSL<br>BCD2A<br>B2BCD<br>INT.D | ASMB,R,B,L,T,C |
| 1 | Y0221 LIBRARY TAPE<br>KROSS<br>TTY1<br>HASSL<br>BCD2A<br>B2BCD<br>INT.D | |

NOTE: No control statement should be on Source Tape if Assembly Language

CONTRIBUTED PROGRAM DOCUMENTATION FORM

## 4.0 USER PROCEDURES (Cont.)

### 4.2 Load and Run Instructions

1) FORM AN OVERLAY, ENTERING "Ø221" AS A NUMBER
2) FORCE LOAD "YØ221 LIBRARY"
3) LIBRARY LOAD "BCS RELOCATABLE LIBRARY"
4) END OVERLAY GENERATION
5) LOAD OVERLAY OVER SYSTEM AND RUN
6) CALL Y221 TO START

### 4.3 DIAGNOSTIC MESSAGES OR ADDITIONAL EXITS

| | |
|---|---|
| TT WHAT? | UNDEFINED DATA TYPE (BAD COORDINATE CODE) |
| FC WHAT? | CHANNEL # TOO LARGE |

## 5.0 SUBPROGRAM INFORMATION

### 5.1 Entry Points

YØ221

### 5.2 Externals Called

[XX] User Program Utility Library

[ ] FORTRAN II Library

[ ] Fourier Routines

[ ] FORTRAN IV Library

[XX] Others

FLOATING POINT BCS RELOCATABLE
LIBRARY (24250-60001)

CONTRIBUTED PROGRAM DOCUMENTATION FORM

6.0 SPECIAL CONSIDERATIONS

NONE

7.0 EXAMPLE INPUT/OUTPUT (Test Case)

NONE

8.0 LITERATURE REFERENCE

NONE

# HEWLETT-PACKARD FOURIER ANALYZER USER'S GROUP

## CONTRIBUTED PROGRAM DISCLAIMER

[X] INITIAL SUBMISSION [ ] REVISION (PROGRAM NO. )

CLASSIFICATION CODE (a list of classification codes is included on next page)

Y | 0 | 2 | 2 | 1 |
↑ Classification

"KROSS"

TITLE DATA BLOCK CURSOR ROUTINE

CONTRIBUTOR'S NAME DEAN JOHNSON

ORGANIZATION HEWLETT-PACKARD
SANTA CLARA DIVISION

ADDRESS 5301 STEVENS CREEK BLVD.
SANTA CLARA, CALIF. 95050

TELEPHONE (408) 246-4300 EXT-2566

Name and Organization to appear in catalog? [X] Yes [ ] No

May a user contact you directly? [X] Yes [ ] No

MATERIAL SUBMITTED

[X] Documentation [X] Source Program [ ] Test Case

[x] Other LIBRARY

ACKNOWLEDGEMENT AND AGREEMENT:

To the best of my knowledge, this contributed program is free of any proprietary information belonging to any person or organization. I am making this program information available to the HEWLETT-PACKARD Fourier Analyzer User's Group. I hereby agree that Hewlett-Packard may reproduce, publish, and use it, and authorize others to do so without obligations or liability of any kind.

(Signature) (Date)

HEWLETT-PACKARD FOURIER ANALYZER USER'S GROUP

CONTRIBUTED PROGRAM DOCUMENTATION FORM

1.0 IDENTIFICATION
10637-80221

1.1 Program Title
DATA BLOCK CURSOR ROUTINE

1.2 User Program Command Number
Y0221

1.3 Program Language

[ ] FORTRAN II

[X] Assembly Language

[ ] FORTRAN IV

2.0 PROGRAM DESCRIPTION

When called, Y0221 places a cursor across any frequency domain data block displayed ~~(time domain data is ignored).~~ The cursor can be moved left or right, and information - channel #, block #, frequency, and amplitude - can be printed out by pressing display register buttons. The program works on linear or log, rectangular or polar, and on partial displays. If Y0221 is called a second time, the program ends.

The display register controls are:

BIT 7 = FAST RIGHT MOVEMENT
" 8 = STEP RIGHT
" 9 = PRINT INFO
" 10 = HIGH SPEED DISPLAY W/OUT CURSOR
" 11 = ZOOM
" 12 = STEP LEFT
" 13 = FAST LEFT MOVEMENT

Time Data

CHANNEL n.nn OF BLOCK nnnn
TIME = nnnn *10** ±nn SEC
AMPLITUDE = nnnn *10** ± nnn

Freq. Data

← Same
FREQUENCY = nnnn *10** ±nn HZ
← Same

CONTRIBUTED PROGRAM DOCUMENTATION FORM

## 3.0 ENVIRONMENT DESCRIPTION

3.1 Supported Software Requirements

[X] 5451 B Software Release ___

[ ] 5451 __/71 __Software Release ___

[ ] 5451 __/70 __Software Release ___

3.2 Non-Supported Software Requirements

Name and order number of other software packages called by the contributed program:

NONE

3.3 Storage Requirements

FORTRAN, Assembly Language (to be completed by User's Group)

$(2270)_{10}$ without external subprograms

$(442)_{10}$ with external subprograms

$(93)_{10}$ Basepage

3.4 Hardware Requirements

5451B FOURIER SYSTEM

CONTRIBUTED PROGRAM DOCUMENTATION FORM

## 4.0 USER PROCEDURES

### 4.1 Tape Identification

Account for all physical tapes included in this package.

| Tape Number | Contents<br>Programs, tests, data, etc., in order from outside to inside of tape | Control Statement (if any) |
|---|---|---|
| 1 | Y0221 SOURCE TAPE<br>KROSS<br>TTY1<br>HASSL<br>BCD2A<br>B2BCD<br>INT.D | ASMB,R,B,L,<br>T,C |
| 1 | Y0221 LIBRARY TAPE<br>KROSS<br>TTY1<br>HASSL<br>BCD2A<br>B2BCD<br>INT.D | |

NOTE: No control statement should be on Source Tape if Assembly Language

4.0 USER PROCEDURES (Cont.)

4.2 Load and Run Instructions

1) FORM AN OVERLAY, ENTERING "Ø221" AS A NUMBER
2) FORCE LOAD "YØ221 LIBRARY"
3) LIBRARY LOAD "BCS RELOCATABLE LIBRARY"
4) END OVERLAY GENERATION
5) LOAD OVERLAY OVER SYSTEM AND RUN
6) CALL Y221 TO START

4.3 DIAGNOSTIC MESSAGES OR ADDITIONAL EXITS

| | |
|---|---|
| TT WHAT? | UNDEFINED DATA TYPE (BAD COORDINATE CODE) |
| FC WHAT? | CHANNEL # TOO LARGE |

CONTRIBUTED PROGRAM DOCUMENTATION

## 5.0 SUBPROGRAM INFORMATION

### 5.1 Entry Points

YØ221

### 5.2 Externals Called

[X] User Program Utility Library

[ ] FORTRAN II Library

[ ] Fourier Routines

[ ] FORTRAN IV Library

[X] Others

FLOATING POINT BCS RELOCATABLE LIBRARY (24250-60001)

SOUND Velocity Calculation Program. SOUND converts temperature versus depth profile to a sound velocity versus depth profile. A standard salinity of 34PPM is assumed at all depths. The following is the operating instructions for the SOUND program:

| STEP NO. | DESCRIPTION | OBSERVATION |
|---|---|---|
| 1 | Bootstrap load the RTE Software System and start the File Manager | RTE software system is loaded and file management system is operating. |
| 2 | Enter and execute the following FMGR operator commands<br>:LG,1↓<br>:MR,SOUND↓<br>:RU,LOADR,99↓<br>:RU,SOUND↓ | Allocate number tracks to load and go track area. Transfer relocatable program file to LG tracks. Run loader program to load program from LG area. Run SOUND program |
| 3 | Enter a string of depths in feet and temperatures in degrees F separated by a comma. | Each entry is displayed. |
| 4 | Enter a temperature of 0 to terminate the string. | The SOUND velocity table is displayed. |
| 5 | Terminate operation by entering a negative number for depth. | Operation returns to FMGR. |

SUPROF T=00004 IS ON CR00001 USING 00010 BLKS R=0056

```
FTN4,L
      PROGRAM SOUND
C
C
C*************************************************************
C*                                                          *
C*   TITLE:  SOUND VELOCITY CALCULATION                     *
C*                                                          *
C*   MODULE: SOUND       SOURCE: SUPROF     VERSION:  X00   *
C*                                                          *
C*   AUTHOR: J. P. BRETT          DATE:  12 OCT 1977        *
C*                                                          *
C*   MODULE FUNCTION:                                       *
C*                                                          *
C*      THIS PROGRAM CONVERTS TEMPERATURE VS DEPTH PROFILE  *
C*    TO A SOUND VELOCITY VS DEPTH PROFILE.  A STANDARD     *
C*    SALINITY OF 34PPM IS ASSUMED AT ALL DEPTHS.           *
C*                                                          *
C*                                                          *
C*    INPUT PARAMETERS:   DEPTH = DEPTH                     *
C*                         TEMP = TEMPERATURE               *
C*                                                          *
C*    NOTE: A MAXIMUM OF 100 INPUT ENTRIES ARE PERMITTED    *
C*                                                          *
C*    OUTPUT PARAMETERS:   SV = SOUND VELOCITY              *
C*                                                          *
C*************************************************************
C*
C*
C*
      DIMENSION TEMP(100),DEPTH(100),SV(100)
      WRITE(1,33)
```

```
33    FORMAT("VPROF - VELOCITY PROFILE PROGRAM",/,"THIS PROGRAM"
     *" CONVERTS THE TEMPERATURE VS DEPTH",/,"PROFILE INTO THE"
     *" EQUIVALENT SOUND VELOCITY VS DEPTH ",/,"PROFILE ON A"
     *" POINT-BY-POINT BASIS INTERACTIVELY.",/,"TO TERMINATE LOOP"
     *", ENTER A NEGATIVE NUMBER FOR DEPTH."/)
9     WRITE(1,10)
10    FORMAT(//"ENTER A STRING OF DEPTHS IN FEET AND"/
     1"TEMPERATURES IN DEGREES F SEPARATED BY A COMMA."/
     2"ENTER A TEMPERATURE OF 0 TO TERMINATE THE STRING.")
      I=0
      WRITE(1,35)
35    FORMAT(//"DEPTH,TEMP")
50    I=I+1
      READ(1,*)DEPTH(I),TEMP(I)
100   FORMAT(2F10.2)
      IF(DEPTH(1).LT.0.)GOTO14
      IF(TEMP(I).NEQ.0.) GO TO 50
      I=I-1
      DO 70 N=1,I
      TEMP(N)=(TEMP(N)-32.)*5.0/9.0
      DEPTH(N)=DEPTH(N)*0.3048
      SAL = 34.0
      PRES = 1.04 + 0.102506 *(1 + .00528 * SIN(.5846853)**2)*
     1DEPTH(N)+2.524E-7*DEPTH(N)**2
      VT=4.5721*TEMP(N)-4.4532E-2*TEMP(N)**2-2.6045E-4*TEMP(N)**
     $3+7.9851E-6*TEMP(N)**4
      VP = 1.60272E-1*PRES + 1.0268E-5 * PRES**2 + 3.5216E-9 *
     $PRES**3 - 3.3603E-12 * PRES**4
      VS = 1.39799 *(SAL - 35) + 1.69202E-3*(SAL-35)**2
      VSTP=(SAL-35)*(-1.1244E-2*TEMP(N)+7.7711E-7*TEMP(N)**2+
     $7.7016E-5*PRES-1.2943E-7*PRES**2+3.158E-8*PRES*TEMP(N)+
     $1.579E-9*PRES*TEMP(N)**2)+PRES*(-1.8607E-4*TEMP(N)+
     $7.4812E-6*TEMP(N)**2+4.5283E-8*TEMP(N)**3)+PRES**2*
     $(-2.5294E-7*TEMP(N)+1.8563E-9*TEMP(N)**2)+PRES**3*
     $(-1.9646E-10*TEMP(N))
```

```
      VF = 1449.14 + VT + VP + VS + VSTP
   70 SV(N)=VF*3.280833
      DO 80 N=1,I
      DEPTH(N)=DEPTH(N)*3.280833
      TEMP(N)=TEMP(N)*9.0/5.0+32.0
   80 CONTINUE
      WRITE(1,13)
   13 FORMAT(20X,"TEMPERATURE     DEPTH     SOUND VELOCITY",/)
      WRITE(1,12)(TEMP(N),DEPTH(N),SV(N),N=1,I)
   12 FORMAT(23X,F4.1,7X,F7.1,7X,F6.1)
      GO TO 9
   14 END
      END$
```

## 5.5 OLAAS SYSTEM/CORELOAD TRANSFER FILES

A powerful feature of the RTE System is its ability to run from "Transfer Files". A transfer file is an ASCII file containing a sequence of File Manager directives. Using transfer files, the basic system and the various coreloads are generated. The following are the Transfer Files used in the OLAAS System:

| | | |
|---|---|---|
| MPTR | - | Basic System File |
| MPTRØ | - | Coreload Ø File |
| RTTR | - | Basic System for Coreloads Ø & 2 File |
| MPTR1 | - | Coreload 1 File |
| MPTR2 | - | Coreload 2 File |
| MPTR5 | - | Coreload 5 File |
| MPTR6 | - | Coreload 6 File |
| MPTR7 | - | Coreload 7 File |

```
MPTR   T=00004 IS ON CR00001 USING 00002 BLKS R=0013

:RU,CSORT,BR,1C,SC,,1
:PU,PCSJOB
:PU,MPSYS
:PU,MPSNST
:PU,SOGJOB
:PK,-2
:DU,RAFPC,PCSINP
:DU,CSORTC,CMMAND
:RP,FXL2
:RU,FXL
:RP,,FXL2
:TR
      MPTR TRANSFER FILE CREATES MPSYS, THE BASIC BCS AND
    FOURIER SYSTEMS (32K, MASS STORE), USING COMMAND FILE,
    BR1CSC, AND PCS INPUT FILE, RAFPC.
    COMMAND FILE, BR1CSC, IS AN EXPANDED VERSION OF RAFCM.

:
```

```
MPTRO  T=00004 IS ON CR00001 USING 00001 BLKS R=0008

:PU,#DCP
:ST,#Y0003,#DCP
:DU,#CDUMP,#DCP,,2,1
:DU,#CLOAD,#DCP,,2,1
:DU,#TYPI1,#DCP,,2,1
:TR,RTTR
:TR
         MPTR0 TRANSFER FILE CREATES #DCP, WHICH CONTAINS THE
      RELOCATABLE INPUTS FOR GENERATION OF CORELOAD 0, THE OLAAS
      EXECUTIVE PROGRAM.

:
```

```
RTTR    T=00004 [illegible] UP00001 USING 00002 BLKS R=0011

:PU,PCSJOB
:PU,DCPSYS
:PK,-2
:DU,RTPC,PCSINP
:DU,RTCM,CMMAND
:RP,FXL2
:RU,FXL
:RP,,FXL2
:TR
      RTTR TRANSFER FILE CREATES DCPSYS, THE BASIC BCS SYSTEM
    USED IN THE GENERATION OF CORELOAD 0 & 2, USING $DCP AND
    THE PCS INPUT FILE, RTPC.

:
```

```
MPTR1  T=00004 IS ON CR00001 USING 00001 BLKS R=0004

0001  :PU,RAF
0002  :PK,-2
0003  :DU,MPCM1,CMMAND
0004  :RU,FXL3
0005  :TR
0006
0007        MPTR1 TRANSFER FILE GENERATES CORELOAD 1, ANALYSIS &
0008      SYSTEM EXECUTIVE SUBROUTINE, USING COMMAND FILE, MPCM1.

:
```

```
MPTR2  T=00004 IS ON CR00001 USING 00002 BLKS R=0012

:PU,#DCP
:ST,#Y0001,#DCP
:DU,#LKTBG,#DCP,,2,1
:DU,#LINK6,#DCP,,2,1
:DU,#MPINP,#DCP,,2,1
:DU,#CDUMP,#DCP,,2,1
:DU,#CLOAD,#DCP,,2,1
:DU,INPTO,#DCP,,2,1
:DU,#TYPI1,#DCP,,2,1
:DU,#TAT1,#DCP,,2,1
:TR,RTTR
:TR
        MPTR2 TRANSFER FILE CREATES #DCP, WHICH CONTAINS THE
     RELOCATABLE INPUTS FOR GENERATION OF CORELOAD 2, THE
     OLAAS RAW DATA COLLECTION SUBROUTINE.

:
```

```
MPTR5  T=00004 IS ON CR00001 USING 00001 BLKS R=0004

:PU,RAF
:PK,-2
:DU,MPCM5,CMMAND
:RU,FXL3
:TR
      MPTR5 TRANSFER FILE GENERATES CORELOAD 5, OLAAS USER
   PROGRAM, USING THE COMMAND FILE, MPCM5.

:
```

```
MPTR6   T=00004 IS ON CR00001 USING 00001 BLKS R=0005

0001   :PU,RAF
0002   :PK,-2
0003   :DU,MPCM6,CMMAND
0004   :RU,FXL3
0005   :TR
0006          MPTR6 TRANSFER FILE GENERATES CORELOAD 6, FOURIER
0007       ANALYZER GRAPHICS, USING COMMAND FILE, MPCM6.

:
```

```
MPTR7  T=00004 IS ON CR00001 USING 00001 BLKS R=0005

:PU,RAF
:PK,-2
:DU,MPCM7,CMMAND
:RU,FXL3
:TR
      MPTR7 TRANSFER FILE GENERATES CORELOAD 7, OLAAS PLOT
    DISPLAY SUBROUTINE, USING COMMAND FILE, MPCM7.

:
```

## 5.6 OLAAS SYSTEM/CORELOAD COMMAND FILES

A Command File is an ASCII file containing a sequence of FXL commands. FXL, the Fourier Cross Loader, is a package which operates in the RTE Software System to generate automatically the Fourier System, SOG, BCS and Overlays, as well as the listings that accompany them. The information generated is stored in RTE files for later output. Included with FXL is the program COPYF (Copy Files), which can be used to store RTE-generated systems and overlays directly into Fourier File 6 on the Fourier disc. This transfers software from RTE files to the Fourier Analyzer disc. The following are the Command Files used in the OLAAS System:

BR1CSC - Basic System File
MPCM1 - Coreload 1 File
MPCM5 - Coreload 5 File
MPCM6 - Coreload 6 File
MPCM7 - Coreload 7 File

BR1CSC T=00003 IS ON CR00001 USING 00014 BLKS R=0000

```
**
.       BR1CSC COMMAND FILE IS FOR THE BUNKER RAMO 5451B
        WHICH CONTAINS PROVISION FOR THE FOLLOWING:
        VARIABLE PARAMETERS
              7900A  (I0=17)
        THE DUMMY DRIVERS IN THE SYSTEM ARE:
              D.06, D.10, D.53, D.56

        THE DESIGN OF THIS SYSTEM HAS BEEN DONE WITH FLEXIBILITY OF
        USE IN MIND.   USING THIS SYSTEM AS A BASE, MOST OF THE
        STANDARD OPTINS CAN BE IMPLEMENTED BY MAKING APPROPRIATE
        OVERLAYS.

        THIS COMMAND FILE GOES WITH THE PCS ANSWER FILE,  RAFPC.
        GENERATED ARE PCSJOB (BCS SYSTEM TAPE) AND MPSYS (THE BASIC
        FOURIER CORELOAD).    THE SNAPSHOT FILE IS MPSNST.

        THE TRANSFER FILE IS MPTR.
                                            W. BOWEN   11/3/77

        PCS OPERATIONS
A,PCSJOB
        DBOOT
F,A4501D
Y,**
        IOC
F,B3000C
Y,IOC
        REST OF PCS LIB
F,B3000C
Y,D.00
Y,LOADR
```

```
        END OF PCS
$$
A,MPSYS
F,#USTRT
B,**
     ************************************
        BCS OPERATIONS
                              CSORT+
        FLIB1 LIBRARY TO BASEPAGE
F,B3052B
B,INS
F,B3052B
Y,CONST
        FLIB2 LIBRARY
F,B3053D
Y,RUNS
F,B3053D
Y,WCS36
                              CSORT-
                              CSORT+
F,B3053D
Y,CNTRL
F,B3053D
Y,XEQ
F,B3053D
Y,WHAT
F,B3053D
Y,INTCH
F,B3053D
Y,ADCPG
F,B3053D
Y,CIO
F,B3053D
Y,CONV
F,B3053D
```

```
0068  Y,EXEC
0069  F,B3053D
0070  Y,CHKN
0071  F,B3053D
0072  Y,WRBUF
0073  F,B3053D
0074  Y,CPLXM
0075  F,B3053D
0076  Y,LOGA
0077  F,B3053D
0078  Y,PROPH
0079  F,B3053D
0080  Y,SKIP
0081  F,B3053D
0082  Y,YCOM
0083  F,B3053D
0084  Y,Y%COM
0085  F,B3053D
0086  Y,EDDEL
0087  F,B3053D
0088  Y,BUF0
0089  F,B3053D
0090  Y,JUMPY
0091  F,B3053D
0092  Y,HIST
0093  F,B3053D
0094  Y,CLEAR
0095  F,B3053D
0096  Y,.MPY.
0097  F,B3053D
0098  Y,.DIV.
0099  F,B3053D
0100  Y,DPSHR
0101  F,B3053D
0102  Y,INTEG
```

```
F,B3053D
Y,DTAD0
F,B3053D
Y,SPEC
F,B3053D
Y,FOUR.
F,B3053D
Y,RANGE
F,B3053D
Y,MULT.
F,B3053D
Y,ADD.
F,B3053D
Y,BDIV
        FLIB3 LIBRARY (NOT LATEST VERSION, WHICH IS D)
F,B3054C
Y,DIS45
F,B3054C
Y,BLANK
F,B3054C
Y,WCS16
F,B3054C
Y,KEYPH
F,B3054C
Y,PRINT
F,B3054C
Y,RDBUF
F,B3054C
Y,KREAD
F,B3054C
Y,ROTAT
F,B3054C
Y,DISP.
F,B3054C
Y,SWRIN
```

```
F,B3054C
Y,BUFIN
F,B3054C
Y,LOCAT
F,B3054C
Y,XSTOR
F,B3054C
Y,BLOK
F,B3054C
Y,HANN
F,B3054C
Y,MAXC
F,B3054C
Y,MAXIM
F,B3054C
Y,COOR
F,B3054C
Y,BDIF
F,B3054C
Y,LECTR
F,B3054C
Y,DPREC
F,B3054C
Y,IF.
F,B3054C
Y,ADSCL
F,B3054C
Y,A%COM
F,B3054C
Y,WCS20
F,B3054C
Y,WCS24
      FLIB4 LIBRARY
F,B3055C
Y,WCS22
```

```
      DISC LIBRARY (NOT LATEST VERSION, WHICH IS G)
F,B3049F
Y,D.34A
F,B3049F
Y,MASS
F,B3049F
Y,.ENTR
F,B3049F
Y,.EAU.
      FMTXX (DISC FORMAT)
F,$FMTX1
L,FMTXX
      ? LIBRARY
F,A3060A
L,D.06
F,A3060A
L,D.10
F,A3060A
L,D.53
F,A3060A
L,D.56
F,A3060A
L,I.10
                    CSORT-
      REMAT
M,72000
F,B3025C
Y,**
R,**
      LIST UNDEFINED SYMBOLS
U,**
      END OF SYSTEM
E,**
      SAVE SNAPSHOT
C,MPSNST
```

```
0208         SOG
0209  A,SOGJOB
0210  M,2010
0211  F,B0009C
0212  Y,**
0213  $$
0214  ::
0215  END

:
```

```
MPCM1   T=00004 I     CR00001 USING 00004 BLKS R=0040

@@
@,RAF
      MPCM1 COMMAND FILE IS FOR GENERATING OVERLAYS FOR THE
    BASIC FOURIER SYSTEM (MPSYS), USING THE SNAPSHOT FILE
    "MPSNST", FOR CORELOAD 1.   THE TRANSFER FILE IS MPTR1.



      OPEN OVERLAY AND SNAPSHOT FILES
T,MPSNST
      ENTER USER PROGRAM NUMBERS
@,1002
      END OF USER PROGRAM LIST
@E
      OPEN USER PROGRAMS
F,Y10020
L,**
F,*RDATA
L,**
F,*TAT1
L,**
F,INPTO
L,**
F,*ASKQ
L,**
F,*TYPI1
L,**
F,*CDUMP
L,**
F,*CLOAD
L,**
F,ISHFTO
```

```
L,**
F,D7500
L,**
F,*DISC
L,**
F,*LINKD
L,**
          END OF RELOCATIBLE USER PROGRAM ENTRY
          LOAD LIBRARIES

          TEKPLOT ROUTINES
F,TEKOC
L,**
F,ERASO
L,**
          USER PROGRAM LIBRARY
F,B3018C
L,**
          BCS FORTRAN IV LIBRARY
F,F4145B
L,**
          BCS FLOATING POINT LIBRARY
F,F4250B
L,**
          LIST UNDEFINED SYMBOLS
U,**
          END
E,01
::
$$

:
```

```
MPCM5   T=00004 IS [illegible] R00001 USING 00006 BLKS R=0092

@@
H.RAF
         MPCM5 COMMAND FILE IS FOR GENERATING OVERLAYS FOR THE
        BASIC FOURIER SYSTEM (MPSYS), USING THE SNAPSHOT FILE
        "MPSNST", FOR CORELOAD 5.    THE TRANSFER FILE IS MPTR5.



         OPEN OVERLAY AND SNAPSHOT FILES
T,MPSNST
         ENTER USER PROGRAM NUMBERS
@,7891
@,7894
@,7896
@,7897
@,7898
@,7899
@,7901
@,7902
@,7904
@,7906
@,7907
         CURSOR PROGRAM
@,0221
         END OF USER PROGRAM LIST
@E
         OPEN USER PROGRAMS
F,Y78910
L,**
F,Y78940
L,**
F,Y78960
```

```
L,**
F,Y78970
L,**
F,Y78980
L,**
F,Y78990
L,**
F,Y79010
L,**
F,Y79020
L,**
F,Y79040
L,**
F,Y79060
L,**
F,Y79070
L,**
F,Y02210
L,**
        OPEN SUBROUTINES CALLED BY USER PROGRAMS
F,BMDATO
L,**
F,DMA10
L,**
F,$RDATA
L,**
F,INPTB
L,**
F,$TYPI
L,**
F,$TAT1
L,**
F,D7500
L,**
F,$RSTAU
```

```
Y,**
          END OF RELOCATIBLE USER PROGRAM ENTRY

          LOAD LIBRARIES

          SCREEN ERASE - MODIFIED TEKTRONIX ROUTINE
F,TEKOC
L,**
F,ERASO
L,**
          USER PROGRAM LIBRARY
F,B3018C
L,**
          BCS FORTRAN IV LIBRARY
F,F4145B
L,**
          BCS FLOATING POINT LIBRARY
F,F4250B
L,**
          LIST UNDEFINED SYMBOLS
U,**
          END
E,01
::
$$

:
```

MPCM6  T=00004 IS ON CP0000: USING 00004 BLKS R=0046

```
@@
A,RAF
      MPCM6 COMMAND FILE IS FOR GENERATING OVERLAYS FOR THE
   BASIC FOURIER SYSTEM (MPSYS), USING THE SNAPSHOT FILE
   "MPSNST", FOR CORELOAD 6.   THE TRANSFER FILE IS MPTR6.


      OPEN OVERLAY AND SNAPSHOT FILES
T,MPSNST
      ENTER USER PROGRAM NUMBERS
@,5804
@,5805
@,5809
@,5810
@,5814
@,5815
@,5816
@,5817
@,5821
      END OF USER PROGRAM LIST
@E

             LOAD LIBRARIES

      GRAPHICS LIBRARY
F,A3015I
L,**
      USER PROGRAM LIBRARY
F,B3018C
L,**
      BCS FLOATING POINT LIBRARY
F,F4250B
```

```
L,**
        GRAPHICS DRIVER D.06
F,A4503A
Y,**
        LIST UNDEFINED SYMBOLS
U,**
        END
E,01
::
$$

:
```

MPCM7 T=00004 IS ON CR00001 USING 00004 BLKS R=0054

```
@@
A,RAF
        MPCM7 COMMAND FILE IS FOR GENERATING OVERLAYS FOR THE
      BASIC FOURIER SYSTEM (BASYS), USING THE SNAPSHOT FILE
      "MPSNST", FOR CORELOAD 7.    THE TRANSFER FILE IS MPTR7.
        OPEN OVERLAY AND SNAPSHOT FILES
T,MPSNST
        ENTER USER PROGRAM NUMBERS
@,1007
@,7909
        END OF USER PROGRAM LIST
@E
        OPEN USER PROGRAMS
F,Y10070
L,**
F,Y79090
L,**
F,$RDATA
L,**
F,$TAT1
L,**
F,INPTO
L,**
F,$TYPI1
L,**
F,$CDUMP
L,**
F,$CLOAD
L,**
```

```
F,*DISC
L,**
F,*LINKD
L,**
F,D7500
L,**
        END OF RELOCATIBLE USER PROGRAM ENTRY
        LOAD LIBRARIES

        TEKPLOT ROUTINES
F,TEKOC
L,**
F,ERASO
L,**
        USER PROGRAM LIBRARY
F,B3018C
L,**
        BCS FORTRAN IV LIBRARY
F,F4145B
L,**
        BCS FLOATING POINT LIBRARY
F,F4250B
L,**
        LIST UNDEFINED SYMBOLS
U,**
        END
E,01
::
$$

:
```

## 5.7 OLAAS SYSTEM/CORELOAD PCS INPUT FILES

The information required by PCS concerning the system I/O configuration is passed on to FXL in an RTE PCS INPUT FILE. A PCS Input File is an ASCII file containing the sequence of all answers required by PCS. The following are the PCS Input files used in the OLAAS System:

| | | |
|---|---|---|
| RAFPC | - | Basic System File |
| RTPC | - | Coreload Ø & 2 File |

```
RHFPC   T=00003 IS ON CR00001 USING 00003 BLKS R=0000

0001  30
0002  77577
0003  0,0
0004  10,D.00
0005  11,D.00
0006  12,D.00
0007  23,D.00
0008  14,D.54
0009  15,D.41
0010  16,D.43
0011  17,D.34
0012  17,D.34,D,U16
0013  17,D.34,D,U17
0014  17,D.34,D,U20
0015  17,D.34,D,U21
0016  17,D.34,D,U22
0017  17,D.34,D,U23
0018  17,D.34,D,U24
0019  17,D.34,D,U25
0020  17,D.34,D,U26
0021  14,D.55
0022  13,D.53
0023  21,D.56
0024  11,D.06
0025  24,D.10
0026  /E
0027  10
0028  10
0029  7
0030  10
0031  7
0032  10
```

```
6,7
4,PFSR
6,106706
7,106707
10,I.00
11,I.00
12,I.00
14,103114
15,I.41
16,I.43
17,102017
20,C.34
21,000000
22,I.00
24,I.10
/E
          RAFPC, THE PCS INPUT FILE USED TO CREATE THE BASIC
      FOURIER SYSTEM, MPSYS.    RAFPC IS USED WITH THE
      TRANSFER FILE, MPTR, AND COMMAND FILE, BR1CSC.

:
```

```
RTPC       00004 IS ON CR00001 USING 00002 BLKS R=0031

0001  30
0002  77577
0003  0.0
0004  10,D.00
0005  11.D.00
0006  12,D.00
0007  23,D.00
0008  /E
0009  10
0010  10
0011  7
0012  10
0013  7
0014  10
0015  6,7
0016  4,102004
0017  6,106706
0018  7,106707
0019  10,I.00
0020  11,I.00
0021  12,I.00
0022  14,103114
0023  15,102015
0024  16,102016
0025  17,102017
0026  20,102020
0027  21,000000
0028  22,102022
0029  23,102023
0030  24,102024
0031  /E
0032          RTPC, PCS INPUT FILE, USED TO CREATE THE BASIC
```

```
FOURIER SYSTEM, DCPSYS.   RTPC IS USED WITH THE
TRANSFER FILES, MPTR0, MPTR2, AND RTTR.

:
```

# APPENDIX
# A

## OLAAS ACCEPTANCE TEST PLAN

# OLAAS ACCEPTANCE TEST PLAN

## 1.0 SCOPE

This report details the proposed Acceptance Test Plan for the On-Line Acoustic Analysis System (OLAAS). The objective of the test is to verify proper operation of OLAAS Processing and Display System as set forth in the OLAAS specification.

## 2.0 OVERVIEW

Bunker Ramo has been engaged in the development of the OLAAS system, which is the computerized data analysis system for the ERAPS Sonobuoy Advanced Development Model. The system is housed in a portable van, and includes the interface to hardware developed under the ERAPS contract as well as the incorporation of hardware and interfaces procured under OLAAS. The development includes a specially written software package, Processing and Display System for the automatic detection analysis of ERAPS data.

## 3.0 TEST SPECIFICATIONS

### 3.1 Test Methods

The Acceptance Test will be made in the OLAAS Van, under the observation and with the interaction of NAVAIRDEVCEN (NADC) personnel. Bunker Ramo personnel will operate the equipment and NADC personnel will be asked to submit arbitrary analysis parameters.

### 3.2 Test Procedures

The Acceptance Tests shall be made in accordance with the detailed procedures described in Section 4.0.

## 4.0 DETAILED TEST PROCEDURE

### 4.1 OLAAS Processing and Display System Software Tests

The OLAAS Processing and Display System software tests will be conducted as follows:

1. Bunker Ramo personnel will set-up the equipment for the tests.
2. NADC personnel will be asked to supply:
   a. Date and time for CW and FM pings.
   b. Changes to the analysis parameters for program verification.
3. System operation shall be according to the OLAAS Processing and Display System Operating Procedures.
4. Raw data (CW or FM) shall be input to the system from the tape recorder using tapes recorded during sea testing.
5. Hard copy prints shall be made of the detection plots, recording and analysis parameters.

### 4.2 Single Ping Recording and Analysis.

1. Using the OLAAS SYSTEM OPERATION PROCEDURE, 2.3, record and analyze a single CW ping using the parameters at time of recording. Verify:
   a. Data is correct for the selected ping.
   b. Date appears on the hard copy prints.
   c. The CW Notch Filtering
2. Using the OLAAS SYSTEM OPERATION PROCEDURE, 2.3, re-analyze the data recorded in step 1 changing only the CW Notch Filter. Verify:
   a. CW Notch Filtering has changed based on the value entered and as noted in step 1.

3. Using the OLAAS SYSTEM OPERATION PROCEDURE, 2.3, request a FOUR PART DISPLAY. Verify:

    a. Data is correct according to step 1.

4. Using the OLAAS SYSTEM OPERATION PROCEDURE, 2.3, record and analyze a single FM ping using the parameters at time of recording. Verify:

    a. Data is correct for the selected ping.
    b. Date appears on the hard copy prints.

5. Repeat step 3 using an FM ping.

6. Using the OLAAS SYSTEM OPERATION PROCEDURE, 2.3, re-analyze the data recorded in step 4 changing selected analysis parameters. Verify:

    a. Data is correct according to step 4.

4.3 Multi-Ping Recording and Analysis

1. Using the OLAAS SYSTEM OPERATION PROCEDURE, 2.3, record and analyze four (4) CW pings using the parameters at time of recording. Verify:
    a. Data is correct for each of the selected pings.

2. Using the OLAAS SYSTEM OPERATION PROCEDURE, 2.3, request a MULTI-PING DISPLAY. Verify:

    a. Data is correct according to step 1.

3. Using the OLAAS SYSTEM OPERATION PROCEDURE, 2.3, request a FOUR PART DISPLAY. Verify:

    a. Data is correct according to step 1 and the ping selected.

4. Using the OLAAS SYSTEM OPERATION PROCEDURE, 2.3, re-analyze the data for a selected ping recorded in step 1 changing selected analysis parameters. Verify:

5. Repeat steps 1 through 4 using FM pings.

## 5.0 TEST SCHEDULE

The OLAAS Acceptance Test shall be made after 30 January 1978 at a time convenient to NADC. The test requires one (1) day to complete.

# APPENDIX

# B

# FMTXX1 PROGRAM LISTING

FMTXX1 T=00004 IS ON CR00001 USING 00052 BLKS R=0409

```
ASMB
     HED A-05451-94194-1 REV.A  NAM FMTXX  ERS & LISTING
     NAM FMTXX
*
*  PURPOSE:    THIS MODULE COMPRISES TABLES AND SUBROUTINES
*              WHICH DEFINE THE 7970 MAGTAPE/7900 DISC STRUCTURE
*              TO THE 5451 MASS-KEYSTROKE PROCESSOR.
*
*  ENTRY POINTS:
*
*  1. FMTXX    THIS 5 WORD/ENTRY TABLE DEFINES THE NUMBER OF
*              RECORDS, AN EQUIPMENT TABLE ENTRY, AND A CORE
*              BUFFER FOR EACH OF THE 10 MASS-KEYSTROKE FILES.
*              THE EQT INFORMATION IS SETUP BY GLOBAL SUBROUTINE
*              MS00.  THE RELATION BETWEEN THE LOGICAL RECORD
*              SIZE AND THE PHYSICAL RECORD SIZE IS ESTABLISHED
*              BY THE MASS SUBROUTINE.
*
*  2. BOOT7    THIS SUBROUTINE CAUSES THE HIGHCORE-RESIDENT MAG-
*              TAPE OR DISC BOOT TO LOCATE AND LOAD AND TRANSFER
*              CONTROL TO ANY CORE-IMAGE CONTAINED ON THE SYSTEM
*              FILE (IE FILE 6).
*
*  3. FMTDD    THIS 8 WORD/ENTRY TABLE DEFINES THE PHYSICAL PAR-
*              TITIONING OF THE 7900 DISC INTO FILES.  THE TABLE
*              IS ONLY USED BY D.34, ITS FORMAT IS DEFINED IN
*              DETAIL THERE.  NOTE THAT MAXIMUM TRANSFER SPEED
*              FOR THE 7900/7901 DISC NOT OBTAINED BY TRANSFERRING
*              DATA FOR A FULL DISC REVOLUTION THEN ACCESSING THE
*              NEXT TRACK, BUT RATHER BY TRANSFERRING FOR ABOUT
*              2/3 OF A REVOLUTION THEN POSITIONING DURING THE
*              LAST 1/3.
```

```
*
*  4. DIFS     THIS IS THE NAME BY WHICH D.34 ACCESS' FMTDD,
*              SEE ABOVE.
*
*  5. FORMT    DUMMY IN CASE MAGTAPE NOT USED.
*
*  6. FMT33    DUMMY IN CASE MAGTAPE NOT USED.
*
*  7. FMT34    DUMMY IN CASE DISC NOT USED.
*
*  8. FMTBB    BUFFER USED BY MASS SUBROUTINE.
*
*  9. XXXXX    NOTE THAT THE MASS STORAGE FILE DESCRIPTIONS
*              INVOLVE A BLOCK OF 10 CONTIGUOUS BCS EQT
*              ENTRIES POINTING TO D.33/D.34.
*
* 10. DACFL    MAKES 5466A & 5465A VERSIONS OF D.33 & D.34 COMPAT.
*
        HED FMTXX,  DISC/MAGTAPE MASS STORAGE DEFINITION
        ENT FMTXX,FMTDD,DIFS,BOOT7
        ENT FORMT,FMT33,FMTBB,FMT34
        ENT DACFL
        EXT BOOT,BOOTT,SPROG,XPROG,DUMPT,MS00
*
        SUP
A       EQU 0
B       EQU 1
**********************************************************************
*  FMTXX,  THIS TABLE IS USED BY MASS, THE MASS-STORAGE KEYSTROKE-
*     PROCESSOR FOR THE 5451 OR 5451/5471 FOURIER PROCESSORS.
*
*  THE TABLE ENTRIES EXIST IN THE FOLLOWING FORMAT  (THE EQT-SPECI
*  INFORMATION IS FILLED IN BY THE MASS PROGRAM WHEN IT'S D.55 ENT
*  POINT IS INVOKED):
*
```

```
*      BSS 1    BITS6-0=EQT UNIT REFERENCE NUMBER
*      BSS 1    NON-INDIRECT DEF TO CORRESPONDING EQT ENTRY
*      DEC N    PHYSICAL RECORD LENGTH
*      DEC M    NUMBER OF RECORDS
*      DEF DEF   DEF TO DEF TO CORE BUFFER USED BY MASS FOR THIS FI
*
*  THE FILES MAY EXIST ON 7970 MAGTAPE (USING D.33A DRIVER) OR ON
*  7900/7901 MOVING-HEAD DISC (USING D.34A DRIVER).  ANY OTHER MED
*  (EG 3030 MAGTAPE, CORE, CASSETTE, ETC) REQUIRES A SPECIAL DRIVE
*  PATTERNED AFTER D.33/D.34, A FILE-POSITION TABLE PATTERNED AFTE
*  FMT33/FMT34, CODING CHANGES TO MASS (RELATING TO 33/34, THE IND
*  FILE, AND IF DESIRED, THE SYSTEM FILE), AND PERHAPS AN AUX TABL
*  LIKE D.34'S DIFS TABLE OR D.33'S EQT CONVENTIONS (SEE BELOW).
*******************************************************************
*
*  AUX D.34 INFO, DIFS TABLE DEFINES PHYSICAL DISC STRUCTURE
*
*  EXAMPLE:
*     DEC 16   LOGICAL FILE NUMBER ARBITRARILY CHOSEN (MATCHES EQT
*     DEC 50   NUMBER OF RECORDS IN FILE
*     DEC 1    DISC SUB-UNIT NUMBER (1 FOR FIRST REMOVABLE DISC)
*     DEC 30   STARTING CYLINDER
*     DEC 0    STARTING SECTOR (48 SECTORS/CYL, 128 WORDS EACH)
*     DEC 8    SECTORS/SECTION (SECTORS USED EACH REC ON ONE CYL)
*     DEC 2    SECTIONS/RECORD (DIFFERENT CYLINDERS EACH RECORD)
*     DEC 20   RECORDS/ROW (# OF RECORDS USING SAME SECTION)
*
*     THE FIRST RECORD OF THIS FILE WOULD APPEAR IN SECTORS 0-7 OF
*     CYLINDERS 30 & 31 FOR A MAX REC SIZE = 128X8X2 = 2048 WORDS.
*     RECORDS 0-19 WOULD GO IN SECTORS 0-7 OF CYLS 30-69.
*     RECORDS 20-39 WOULD GO IN SECTORS 8-15 OF CYLS 30-69.
*     THE FINAL 10 RECORDS WOULD GO IN SECTORS 16-23 OF CYLS 30-49
*
*******************************************************************
*
```

```
        SKP
ENTXX   DEC 5          TABLE WORDS/ENTRY
        DEF *+1        TABLE DEF TO ENTRY0
*
        BSS 2          DUMMY (USED BY MASS,99 ONLY)
        DEC 0,0
        DEF 0
*
        EXT DTAD0      DATA
        BSS 2
        DEC 2048,33
        DEF DTAD0
*
        EXT DTAD0      ADC
        BSS 2
        DEC 4096,0
        DEF DTAD0
*
        EXT STOR0      PROGRAM STACK
        BSS 2
        DEC 336,0
        DEF STOR0
*
        EXT BUF0       ASCII TEXT
        BSS 2
        DEC 36,0
        DEF BUF0
*
        EXT .LBLK      INDEX
        BSS 2
        DEC 21,0
        DEF .LBLK
*
SYS     BSS 2          SYSTEM
        OCT 77700      CORESIZE = LENGTH OF NON-PROTECT
```

```
       DEC 8
       DEF 2
*
       BSS 2          COMMON
       DEC 512,0
       DEF 0
*
       BSS 2          USER8
       DEC 128,0
       DEF 0
*
       BSS 2          DATA BLOCK HEADER
       DEC 128,48
       DEF 0
*******************************************************************
*  BOOT7,  THIS PROGRAM IS USED TO LOAD A PROGRAM FROM THE SYSTEM
*  AND TRANSFER CONTROL TO IT.  DEPENDING ON THE DEVICE ON WHICH T
*  THE SYSTEM FILES ARE STORED, BOOT7 IS REALLY S7900 OR S7970.
*******************************************************************
CORE   ABS 16384      CORESIZE, SET BY BOOT7
SA     OCT 100002     STARTING ADDRESS, SYS FILE CORELOADS
TABL7  DEF BOOTT      DEF TO BOOT'S TABLE
DACFL  NOP
*
BOOT7  NOP            ENTRY: LDA,JSB. (NO RETURN)
       STA RECS
       JSB MS00       MAKE SURE FMTXX IS SETUP
*
       CLC 0          KILL DMA IN CASE ITS BUSY
       STF 6
       STF 7
*
       LDA .BOOT      CALC MAX BOOTABLE ADDRESS
       RSS            (=-DEF BOOT) AND STORE IT
       LDA A,I        IN XPROG FOR BOOT
```

```
        RAL,CLE,SLA,ERA
        JMP *-2
        CMA,INA
        STA XPROG
        CMA,INA
*
        IOR =B7700     CALC SIZE OF MACHINE AND
        AND =B77700   DELETE LOW ORDER 6 BITS
        STA CORE       STORE IN DUMPT FOR DUMP.
        LDB A
        LDA DUMPT
        DST DUMPT
*
        LDA CORE
        CLB            CALC CYLINDERS PER SYS RECORD
        DIV =D6144
        INA
        STA N7900
*
*
        LDB SYS+1      LOOK INTO EQT TO SEE WHAT
        INB            IS THE SYSTEM FILE DEVICE.
        LDA B,I
        ALF,ALF
        AND =B77
        CPA =B33
        JMP S7970          7970.
        CPA =B34
        JMP S7900          7900.
*
S0000   HLT 7B             ELSE, EQT MUST BE CHANGED!
        JMP *-1
*
S7900   BSS 0
        LDA RECS
```

```
      MPY N7900
      AND =B777      SETUP A=0,PLATTER,CYLINDER
      ADA P7900
      JMP GOGO       THEN LOAD AND GO.
*
S7970 BSS 0
      LDA RECS
      MPY N7970
      ALS,ALS
      ALS            SETUP A=0,FILE,REC,NRECS-1
      AND =B7770
      ADA N7970
      ADA =D-1
      ADA F7970
      SZA,RSS        LOAD 1ST RECORD ON THE TAPE ?
      LDA SA            YES, MAKE MBOOT HAPPY
GOGO  LDB SA         AND B=1,STARTINGADDRESS
      DST TABL7,I    THEN STORE IN BOOT TABLE
      JMP SPROG      AND JUMP (NEVER RETURN).
*
*CHARACTERISTICS OF 7970 SYSTEM FILE
N7970 DEC 1          =CORESIZE/RECORDSIZE
F7970 OCT 0          PHYSICAL FILENUMBER
*CHARACTERISTICS OF 7900 OR 7901 SYSTEM FILE
N7900 DEC 2          =CORESIZE/RECORDSIZE(SET BY BOOT7)
P7900 OCT 1          PLATTER (10-9),CYLINDER (8-0)
*
RECS  OCT 0          REC NUMBER IN SYS FILE
.BOOT DEF BOOT
**********1*******************************************************
*  AUX D.34 INFO, DIFS TABLE DEFINES PHYSICAL DISC STRUCTURE.
*  THE BCS EQT TABLE 'SUBUNIT' FIELD IS A POINTER TO THE DIFS
*  'LOGICAL FILE NUMBER'.
******************************************************************
*
```

```
FMTDD DEC 8
DIFS  DEF *+1
*
*
*
      LST               *SUB-UNIT=0 FILE
      DEC 0             -LOGICAL FILE NUMBER
      DEC 203           -NBR OF RECORDS IN FILE
      DEC 0             -SUB-UNIT
      DEC 0             -STARTING CYLINDER
      DEC 0             -STARTING SECTOR
      DEC 48            -SECTORS/SECTION
      DEC 1             -SECTIONS/RECORD
      DEC 203           -RECORDS/ROW
*
*
*
      DEC 1             *SUB-UNIT=1 FILE
      DEC 203
      DEC 1
      DEC 0
      DEC 0
      DEC 48            6144 WORDS
      DEC 1
      DEC 203
*
*                 FILE 1    DATA BLOCKS
*
      OCT 16            LOGICAL UNIT = 16B
      DEC 33           # OF DATA BLOCKS
      DEC 1             REMOVABLE DISC
      DEC 49
      DEC 0
      DEC 16            2048 WORDS/BLOCK
      DEC 1
```

```
0278          DEC 11
0279    *
0280    *          FILE 1        DATA BLOCKS PART 2 (SCRATCH)
0281    *
0282    **    OCT 16
0283    **    DEC 95        # OF DATA BLOCKS
0284    **    DEC 0         FIXED DISC
0285    **    DEC 1
0286    **    DEC 32
0287    **    DEC 16        4096 WORDS/BLOCK
0288    **    DEC 2
0289    **    DEC 95
0290    *
0291    *          FILE 2       ADC THROUGHPUT
0292    *
0293    **    OCT 17
0294    **    DEC 202       # OF THROUGHPUT BLOCKS
0295    **    DEC 0         FIXED DISC
0296    **    DEC 1
0297    **    DEC 0
0298    **    DEC 32        4096 WORDS/BLOCK
0299    **    DEC 1
0300    **    DEC 202
0301    *
0302    *          FILE 3       PROGRAM STACKS
0303    *
0304    **    OCT 20
0305    **    DEC 72        # OF PROGRAM STACKS
0306    **    DEC 1         REMOVABLE DISC
0307    **    DEC 49
0308    **    DEC 34
0309    **    DEC 3         384 WORDS/STACK
0310    **    DEC 1
0311    **    DEC 18
0312    *
```

```
0313  *          FILE 4        ASCII TEXT
0314  *
0315  **      OCT 21
0316  **      DEC 336          # OF ASCII RECORDS
0317  **      DEC 1            REMOVABLE DISC
0318  **      DEC 153
0319  **      DEC 34
0320  **      DEC 1            128 WORDS/RECORD
0321  **      DEC 1
0322  **      DEC 24
0323  *
0324  *          FILE 5        INDEX BLOCKS
0325  *
0326  **      OCT 22
0327  **      DEC 208          # OF INDEX BLOCKS
0328  **      DEC 1            REMOVABLE DISC
0329  **      DEC 49
0330  **      DEC 46
0331  **      DEC 1            128 WORDS/BLOCK
0332  **      DEC 1
0333  **      DEC 104
0334  *
0335  *          FILE 6        CORELOADS
0336  *
0337          OCT 23           LOGICAL UNIT = 23B
0338          DEC 8           # OF  SYSTEM CORELOADS
0339          DEC 1            REMOVABLE DISC
0340          DEC 1            START CYLINDER
0341          DEC 0            START SECTOR
0342          DEC 48           32K/CORELOAD
0343          DEC 6            SECTIONS/RECORD
0344          DEC 8            RECORDS/ROW
0345  *
0346  *          FILE 7        COMMON
0347  *
```

```
**      OCT 24
**      DEC 258          # OF COMMON RECORDS
**      DEC 1            REMOVABLE DISC
**      DEC 67
**      DEC 34
**      DEC 4            512 WORDS/RECORD
**      DEC 1
**      DEC 86
*
*       FILE 8           USER FILE
*
**      OCT 25
**      DEC 1200         # OF USER RECORDS IN PART 1
**      DEC 1            REMOVABLE DISC
**      DEC 177
**      DEC 0
**      DEC 1            128 WORDS/RECORD
**      DEC 1
**      DEC 25
*
*           FILE 9           HEADERS FOR DATA BLOCKS
*
        OCT 26
        DEC 48          # OF DATA BLOCK HEADERS
        DEC 1            REMOVABLE DISC
        DEC 60
        DEC 0
        DEC 1
        DEC 1            128 WORDS/HEADER
        DEC 1
*
*           FILE 9           HEADERS FOR DATA BLOCKS PART 2 (SCRATCH)
*
**      OCT 26
**      DEC 192          # OF SCRATCH DATA HEADERS
```

```
**      DEC 0          FIXED DISC
**      DEC 191
**      DEC 32
**      DEC 1          128 WORDS/HEADER
**      DEC 1
**      DEC 12
*
        DEC -1         *END OF DISC FILE
********************************************************************
*  AUX D.33 INFO, DEFINES PHYSICAL MAGTAPE STRUCTURE.
*  STRUCTURE IS CONTAINED IN THE 'DMA' AND 'SUBUNIT' FIELDS OF THE
*  APPROPRIATE BCS EQT ENTRIES.  THE PCS-TIME TYPINS ARE OF THE FO
*
        CC,D.33[,D],U@@$
*
*  WHERE CC IS THE (LOWER NUMBERED) SELECT CODE OF THE MAGTAPE, AN
*  $ SPECIFIES THE DAISY-CHAIN UNIT CONTAINING THE FILE, @@ IS THE
*  PHYSICAL FILE NUMBER (0,1,2,...) OF THE FILE, AND [,D] IS SPECI
*  IF THE FILE IS FORMATTED (IE, REWRITABLE).
********************************************************************
FMT33 BSS 0
FMT34 DEC 1
      DEF FMTBB
FMTBB BSS 21
FORMT NOP              DUMMY MAGTAPE FORMT
      JMP FORMT,I
      UNS
      END

:
```

# APPENDIX

# C

# DISC DIRECTORY (LOWER DISC) FOR RTE SYSTEM DISC

DL.,1
CR=00001
ILAB=SYSTEM NXTR=0194 NXSEC=018 #SEC/TR=096 LAST TR= 0202 #DR TR=01

| NAME | TYPE | #BLKS/LU | SCODE | TRACK | SEC | OPEN TO |
|---|---|---|---|---|---|---|
| B3014C | 00005 | 00009 | 00000 | 0060 | 000 | |
| B3022D | 00005 | 00079 | 00000 | 0060 | 018 | |
| B3023F | 00005 | 00084 | 00000 | 0061 | 080 | |
| B3025B | 00005 | 00023 | 00000 | 0063 | 056 | |
| A4500A | 00005 | 00001 | 00000 | 0064 | 006 | |
| F0005B | 00005 | 00004 | 00000 | 0064 | 008 | |
| F0017C | 00005 | 00007 | 00000 | 0064 | 016 | |
| A3060A | 00003 | 00045 | 00000 | 0064 | 030 | |
| F4250B | 00005 | 00082 | 00000 | 0065 | 024 | |
| B3018C | 00005 | 00021 | 00000 | 0066 | 092 | |
| RAFPC | 00003 | 00003 | 00000 | 0067 | 038 | |
| A4501D | 00005 | 00003 | 00000 | 0067 | 044 | |
| B3052B | 00005 | 00004 | 00000 | 0067 | 050 | |
| B3025C | 00005 | 00023 | 00000 | 0067 | 058 | |
| B3055C | 00005 | 00007 | 00000 | 0068 | 008 | |
| B3050B | 00005 | 00005 | 00000 | 0068 | 022 | |
| B0009C | 00005 | 00008 | 00000 | 0068 | 032 | |
| OVRTR | 00003 | 00001 | 00000 | 0068 | 048 | |
| Y7892S | 00003 | 00002 | 00000 | 0068 | 050 | |
| OVRTRT | 00004 | 00001 | 00000 | 0068 | 054 | |
| F4145B | 00005 | 00107 | 00000 | 0068 | 056 | |
| Y7892O | 00005 | 00003 | 00000 | 0070 | 078 | |
| Y7893S | 00003 | 00003 | 00000 | 0070 | 084 | |
| Y7893O | 00005 | 00006 | 00000 | 0070 | 090 | |
| EOF | 00004 | 00001 | 00000 | 0071 | 006 | |
| ERDATS | 00004 | 00008 | 00000 | 0071 | 008 | |
| ERDATO | 00005 | 00002 | 00000 | 0071 | 024 | |
| NIBBLS | 00003 | 00013 | 00000 | 0071 | 028 | |
| Y7895O | 00005 | 00005 | 00000 | 0071 | 054 | |

| | | | | | |
|---|---|---|---|---|---|
| A3016F | 00005 | 00031 | 00000 | 0071 | 064 |
| A3015I | 00005 | 00075 | 00000 | 0072 | 030 |
| A4503A | 00005 | 00008 | 00000 | 0073 | 084 |
| LINK6 | 00004 | 00002 | 00000 | 0074 | 004 |
| LKTBG | 00004 | 00002 | 00000 | 0074 | 008 |
| #LKTBG | 00005 | 00001 | 00000 | 0074 | 012 |
| #LINK6 | 00005 | 00001 | 00000 | 0074 | 014 |
| RTTR | 00004 | 00002 | 00000 | 0074 | 016 |
| RTPC | 00004 | 00002 | 00000 | 0074 | 020 |
| &DDISC | 00004 | 00002 | 00000 | 0074 | 024 |
| BEAM1 | 00004 | 00015 | 00000 | 0074 | 028 |
| Y7898X | 00003 | 00002 | 00000 | 0074 | 058 |
| BEAM3 | 00004 | 00014 | 00000 | 0074 | 062 |
| Y0004 | 00004 | 00002 | 00000 | 0074 | 090 |
| TAT | 00004 | 00003 | 00000 | 0074 | 094 |
| DRPTR | 00004 | 00001 | 00000 | 0075 | 004 |
| DMA1S | 00004 | 00008 | 00000 | 0075 | 006 |
| Y7891Q | 00004 | 00006 | 00000 | 0075 | 022 |
| DMA1O | 00005 | 00002 | 00000 | 0075 | 034 |
| Y7896O | 00005 | 00006 | 00000 | 0075 | 038 |
| Y7899S | 00003 | 00004 | 00000 | 0075 | 050 |
| Y7899O | 00005 | 00002 | 00000 | 0075 | 058 |
| RTCM | 00004 | 00004 | 00000 | 0075 | 062 |
| Y9200O | 00005 | 00033 | 00000 | 0075 | 070 |
| PIN | 00004 | 00004 | 00000 | 0076 | 040 |
| Y7900S | 00003 | 00002 | 00000 | 0076 | 048 |
| Y7900O | 00005 | 00002 | 00000 | 0076 | 052 |
| Y7891X | 00004 | 00004 | 00000 | 0076 | 056 |
| D1200 | 00003 | 00018 | 00000 | 0076 | 064 |
| Y7897T | 00004 | 00011 | 00000 | 0077 | 004 |
| DO1 | 00005 | 00008 | 00000 | 0077 | 026 |
| TYPI | 00004 | 00022 | 00000 | 0077 | 042 |
| FMTXX | 00003 | 00052 | 00000 | 0077 | 086 |
| Y0005 | 00004 | 00043 | 00000 | 0078 | 094 |
| B3000C | 00005 | 00054 | 00000 | 0079 | 084 |

| | | | | | |
|---|---|---|---|---|---|
| B3053D | 00005 | 00088 | 00000 | 0081 | 000 |
| B3054C | 00005 | 00074 | 00000 | 0082 | 080 |
| B3049F | 00005 | 00056 | 00000 | 0084 | 036 |
| Y9200 | 00005 | 00033 | 00000 | 0085 | 052 |
| OVRCM | 00003 | 00003 | 00000 | 0086 | 022 |
| Y02210 | 00005 | 00029 | 00000 | 0086 | 028 |
| Y7895S | 00003 | 00006 | 00000 | 0086 | 086 |
| BEAM | 00003 | 00014 | 00000 | 0087 | 002 |
| Y7897X | 00003 | 00010 | 00000 | 0087 | 030 |
| DOUG | 00004 | 00008 | 00000 | 0087 | 050 |
| STPAR | 00004 | 00002 | 00000 | 0087 | 066 |
| #STPAR | 00005 | 00002 | 00000 | 0087 | 070 |
| #Y0004 | 00005 | 00003 | 00000 | 0087 | 074 |
| Y79030 | 00005 | 00003 | 00000 | 0087 | 080 |
| Y79010 | 00005 | 00003 | 00000 | 0087 | 086 |
| RAFCM | 00003 | 00007 | 00000 | 0087 | 092 |
| NIBBL | 00003 | 00013 | 00000 | 0088 | 010 |
| CLOAD | 00004 | 00004 | 00000 | 0088 | 036 |
| CDUMP | 00004 | 00004 | 00000 | 0088 | 044 |
| #CLOAD | 00005 | 00001 | 00000 | 0088 | 052 |
| #CDUMP | 00005 | 00001 | 00000 | 0088 | 054 |
| #TEST | 00005 | 00005 | 00000 | 0088 | 056 |
| D750S | 00004 | 00002 | 00000 | 0088 | 066 |
| ERASO | 00005 | 00001 | 00000 | 0088 | 070 |
| #Y7905 | 00005 | 00003 | 00000 | 0088 | 072 |
| OVRC1 | 00004 | 00004 | 00000 | 0088 | 078 |
| OVRTR1 | 00004 | 00001 | 00000 | 0088 | 086 |
| EXTR | 00004 | 00002 | 00000 | 0088 | 088 |
| Y0003 | 00004 | 00004 | 00000 | 0088 | 092 |
| Y7905S | 00004 | 00003 | 00000 | 0089 | 004 |
| Y79050 | 00005 | 00001 | 00000 | 0089 | 010 |
| RDATA | 00004 | 00041 | 00000 | 0089 | 012 |
| ARR10 | 00005 | 00013 | 00000 | 0089 | 094 |
| BMDATS | 00004 | 00008 | 00000 | 0090 | 024 |
| Y7894S | 00004 | 00009 | 00000 | 0090 | 040 |

| | | | | | |
|---|---|---|---|---|---|
| Y78940 | 00005 | 00004 | 00000 | 0090 | 058 |
| BMDATO | 00005 | 00002 | 00000 | 0090 | 066 |
| Y1003S | 00004 | 00002 | 00000 | 0090 | 070 |
| TEKOO | 00005 | 00004 | 00000 | 0090 | 074 |
| INPTS | 00004 | 00004 | 00000 | 0090 | 082 |
| Y10030 | 00005 | 00002 | 00000 | 0090 | 090 |
| #RSTAV | 00005 | 00005 | 00000 | 0090 | 094 |
| RSTAV | 00004 | 00029 | 00000 | 0091 | 008 |
| #Y0005 | 00005 | 00007 | 00000 | 0091 | 066 |
| Y7909S | 00004 | 00005 | 00000 | 0091 | 080 |
| OVRC7 | 00004 | 00004 | 00000 | 0091 | 090 |
| OVRTR7 | 00004 | 00001 | 00000 | 0092 | 002 |
| Y8891S | 00004 | 00004 | 00000 | 0092 | 004 |
| Y0006 | 00004 | 00035 | 00000 | 0092 | 012 |
| CHARO | 00005 | 00002 | 00000 | 0092 | 082 |
| DMA2S | 00004 | 00008 | 00000 | 0092 | 086 |
| DMA20 | 00005 | 00002 | 00000 | 0093 | 006 |
| INPTA | 00004 | 00004 | 00000 | 0093 | 010 |
| INPTB | 00005 | 00002 | 00000 | 0093 | 018 |
| INPTO | 00005 | 00002 | 00000 | 0093 | 022 |
| Y7906A | 00004 | 00014 | 00000 | 0093 | 026 |
| Y79090 | 00005 | 00003 | 00000 | 0093 | 054 |
| OVRTR3 | 00004 | 00001 | 00000 | 0093 | 060 |
| Y7910 | 00005 | 00006 | 00000 | 0093 | 062 |
| Y7907R | 00004 | 00018 | 00000 | 0093 | 074 |
| Y79100 | 00005 | 00006 | 00000 | 0094 | 014 |
| Y7908S | 00004 | 00003 | 00000 | 0094 | 026 |
| Y79080 | 00005 | 00002 | 00000 | 0094 | 032 |
| Y0003S | 00004 | 00004 | 00000 | 0094 | 036 |
| ISHFTS | 00003 | 00003 | 00000 | 0094 | 044 |
| ISHFTO | 00005 | 00001 | 00000 | 0094 | 050 |
| Y79020 | 00005 | 00008 | 00000 | 0094 | 052 |
| Y1002T | 00004 | 00075 | 00000 | 0094 | 068 |
| Y7906R | 00004 | 00015 | 00000 | 0096 | 026 |
| Y7904S | 00004 | 00007 | 00000 | 0096 | 056 |

| | | | | | | |
|---|---|---|---|---|---|---|
| OVDEU | 00004 | 00006 | 00000 | 0096 | 070 | |
| OVRDEU | 00004 | 00001 | 00000 | 0096 | 082 | |
| Y7912S | 00004 | 00004 | 00000 | 0096 | 084 | |
| Y79120 | 00005 | 00004 | 00000 | 0096 | 092 | |
| Y79VPS | 00004 | 00007 | 00000 | 0097 | 004 | |
| VELOCP | 00005 | 00008 | 00000 | 0097 | 018 | |
| Y7907A | 00004 | 00021 | 00000 | 0097 | 034 | |
| ROMV | 00004 | 00003 | 00000 | 0097 | 076 | |
| ROMVO | 00005 | 00003 | 00000 | 0097 | 082 | |
| VELOCO | 00005 | 00010 | 00000 | 0097 | 088 | |
| VTESTS | 00004 | 00011 | 00000 | 0098 | 012 | |
| FXL | 00006 | 00054 | 00000 | 0098 | 034 | |
| FXL2 | 00006 | 00052 | 00000 | 0099 | 046 | |
| FXL3 | 00006 | 00055 | 00000 | 0100 | 054 | |
| COPYF | 00006 | 00022 | 00000 | 0101 | 068 | |
| BEAMF | 00006 | 00045 | 00000 | 0102 | 016 | |
| PATRN | 00006 | 00037 | 00000 | 0103 | 010 | |
| CALC1 | 00006 | 00026 | 00000 | 0103 | 084 | |
| $LINKD | 00005 | 00001 | 00000 | 0104 | 040 | |
| BCSJOB | 00007 | 00024 | 00000 | 0104 | 042 | |
| BCSJOB | 00007 | 00024 | 00000 | 0104 | 090 | +001 |
| BCSJOB | 00007 | 00024 | 00000 | 0105 | 042 | +002 |
| BCSJOB | 00007 | 00024 | 00000 | 0105 | 090 | +003 |
| BCSJOB | 00007 | 00024 | 00000 | 0106 | 042 | +004 |
| BCSJOB | 00007 | 00024 | 00000 | 0106 | 090 | +005 |
| BCSJOB | 00007 | 00024 | 00000 | 0107 | 042 | +006 |
| BCSJOB | 00007 | 00024 | 00000 | 0107 | 090 | +007 |
| BCSJOB | 00007 | 00024 | 00000 | 0108 | 042 | +008 |
| BCSJOB | 00007 | 00024 | 00000 | 0108 | 090 | +009 |
| RTGEN | 00003 | 00003 | 00000 | 0109 | 042 | |
| BTEST | 00006 | 00097 | 00000 | 0109 | 048 | |
| Y7907 | 00004 | 00002 | 00000 | 0111 | 050 | |
| $TYPI | 00005 | 00004 | 00000 | 0111 | 054 | |
| Y1111S | 00004 | 00066 | 00000 | 0111 | 062 | |
| %DDISC | 00005 | 00002 | 00000 | 0113 | 002 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| DDISC | 00006 | 00025 | 00000 | 0113 | 006 | |
| INPUT | 00004 | 00006 | 00000 | 0113 | 056 | |
| #INPUT | 00005 | 00005 | 00000 | 0113 | 068 | |
| %CSORT | 00005 | 00072 | 00000 | 0113 | 078 | |
| Y7911S | 00004 | 00041 | 00000 | 0115 | 030 | |
| OVCSC1 | 00004 | 00006 | 00000 | 0116 | 016 | |
| OVCST1 | 00004 | 00001 | 00000 | 0116 | 028 | |
| OVCSC5 | 00004 | 00006 | 00000 | 0116 | 030 | |
| OVCST5 | 00004 | 00001 | 00000 | 0116 | 042 | |
| OVRC5 | 00004 | 00006 | 00000 | 0116 | 044 | |
| Y7901S | 00004 | 00006 | 00000 | 0116 | 056 | |
| Y7903S | 00003 | 00004 | 00000 | 0116 | 068 | |
| CHOUT | 00003 | 00028 | 00000 | 0116 | 076 | |
| ARR1 | 00004 | 00014 | 00000 | 0117 | 036 | |
| TEKS | 00003 | 00056 | 00000 | 0117 | 064 | |
| Y7907S | 00004 | 00020 | 00000 | 0118 | 080 | |
| Y7910S | 00004 | 00007 | 00000 | 0119 | 024 | |
| Y7902S | 00004 | 00009 | 00000 | 0119 | 038 | |
| VELOCS | 00004 | 00008 | 00000 | 0119 | 056 | |
| PLMOD1 | 00004 | 00047 | 00000 | 0119 | 072 | |
| PCSINP | 00003 | 00002 | 00000 | 0120 | 070 | |
| CMMAND | 00004 | 00004 | 00000 | 0120 | 074 | |
| BTESTS | 00003 | 00010 | 00000 | 0120 | 082 | |
| CSORT | 00006 | 00092 | 00000 | 0121 | 006 | |
| CMMAND | 00004 | 00004 | 00000 | 0122 | 094 | +001 |
| CMMAND | 00004 | 00004 | 00000 | 0123 | 006 | +002 |
| FMTXX1 | 00004 | 00052 | 00000 | 0123 | 014 | |
| #FMTX1 | 00005 | 00004 | 00000 | 0124 | 022 | |
| FMTXX2 | 00004 | 00052 | 00000 | 0124 | 030 | |
| #FMTX2 | 00005 | 00004 | 00000 | 0125 | 038 | |
| TAT1 | 00004 | 00003 | 00000 | 0125 | 046 | |
| TAT2 | 00004 | 00003 | 00000 | 0125 | 052 | |
| #TAT2 | 00005 | 00001 | 00000 | 0125 | 058 | |
| BR1TR1 | 00004 | 00002 | 00000 | 0125 | 060 | |
| PCSINP | 00003 | 00002 | 00000 | 0125 | 064 | +001 |

| | | | | | | |
|---|---|---|---|---|---|---|
| #RDATA | 00005 | 00006 | 00000 | 0125 | 068 | |
| BR1CSC | 00003 | 00014 | 00000 | 0125 | 080 | |
| CMMAND | 00004 | 00004 | 00000 | 0126 | 012 | +003 |
| CMMAND | 00004 | 00004 | 00000 | 0126 | 020 | +004 |
| RAFMP | 00003 | 00007 | 00000 | 0126 | 028 | |
| OVRC6 | 00004 | 00006 | 00000 | 0126 | 042 | |
| OVRTR6 | 00004 | 00001 | 00000 | 0126 | 054 | |
| VPROF | 00005 | 00010 | 00000 | 0126 | 056 | |
| Y7913S | 00004 | 00015 | 00000 | 0126 | 076 | |
| Y79070 | 00005 | 00014 | 00000 | 0127 | 010 | |
| Y7906 | 00005 | 00010 | 00000 | 0127 | 038 | |
| Y79130 | 00005 | 00014 | 00000 | 0127 | 058 | |
| SOUND | 00005 | 00014 | 00000 | 0127 | 086 | |
| Y79060 | 00005 | 00009 | 00000 | 0128 | 018 | |
| #TAT1 | 00005 | 00001 | 00000 | 0128 | 036 | |
| Y79040 | 00005 | 00006 | 00000 | 0128 | 038 | |
| #TAT0 | 00005 | 00001 | 00000 | 0128 | 050 | |
| Y10020 | 00005 | 00049 | 00000 | 0128 | 052 | |
| DISC | 00003 | 00024 | 00000 | 0129 | 054 | |
| #Y0010 | 00005 | 00004 | 00000 | 0130 | 006 | |
| USTRT | 00004 | 00002 | 00000 | 0130 | 014 | |
| #USTRT | 00005 | 00001 | 00000 | 0130 | 018 | |
| MPTR | 00004 | 00002 | 00000 | 0130 | 020 | |
| #FPGRD | 00005 | 00008 | 00000 | 0130 | 024 | |
| OLCMCS | 00003 | 00014 | 00000 | 0130 | 040 | |
| RDTR | 00004 | 00002 | 00000 | 0130 | 068 | |
| #DISC | 00005 | 00004 | 00000 | 0130 | 072 | |
| MPTR7 | 00004 | 00001 | 00000 | 0130 | 080 | |
| BSNST2 | 00003 | 00024 | 00000 | 0130 | 082 | |
| BSNST2 | 00003 | 00024 | 00000 | 0131 | 034 | +001 |
| MPCM7 | 00004 | 00004 | 00000 | 0131 | 082 | |
| MPCM6 | 00004 | 00004 | 00000 | 0131 | 090 | |
| MPCM5 | 00004 | 00006 | 00000 | 0132 | 002 | |
| OVRTR5 | 00004 | 00001 | 00000 | 0132 | 014 | |
| MPTR5 | 00004 | 00001 | 00000 | 0132 | 016 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| MPTR6 | 00004 | 00001 | 00000 | 0132 | 018 | |
| #TAT | 00005 | 00001 | 00000 | 0132 | 020 | |
| RAFTR | 00004 | 00002 | 00000 | 0132 | 022 | |
| MPTR1 | 00004 | 00001 | 00000 | 0132 | 026 | |
| MPCM1 | 00004 | 00004 | 00000 | 0132 | 028 | |
| RDCPTR | 00004 | 00002 | 00000 | 0132 | 036 | |
| RDCPCM | 00004 | 00004 | 00000 | 0132 | 040 | |
| Y0011 | 00003 | 00034 | 00000 | 0132 | 048 | |
| ANMOD5 | 00004 | 00083 | 00000 | 0133 | 020 | |
| PLMOD5 | 00004 | 00047 | 00000 | 0134 | 090 | |
| SUPROF | 00004 | 00016 | 00000 | 0135 | 088 | |
| Y7896S | 00004 | 00015 | 00000 | 0136 | 024 | |
| PGMHED | 00004 | 00007 | 00000 | 0136 | 054 | |
| Y7906S | 00004 | 00017 | 00000 | 0136 | 068 | |
| ANMOD7 | 00004 | 00089 | 00000 | 0137 | 006 | |
| MP003A | 00004 | 00011 | 00000 | 0138 | 088 | |
| MP1007 | 00004 | 00110 | 00000 | 0139 | 014 | |
| D7500 | 00005 | 00001 | 00000 | 0141 | 042 | |
| DCPTR | 00004 | 00002 | 00000 | 0141 | 044 | |
| MPTR2 | 00004 | 00002 | 00000 | 0141 | 048 | |
| TYPI1S | 00004 | 00050 | 00000 | 0141 | 052 | |
| #TYPI1 | 00005 | 00006 | 00000 | 0142 | 056 | |
| MPTR0 | 00004 | 00001 | 00000 | 0142 | 068 | |
| MP007A | 00004 | 00138 | 00000 | 0142 | 070 | |
| MPTR1C | 00004 | 00001 | 00000 | 0145 | 058 | |
| MPCM1C | 00004 | 00007 | 00000 | 0145 | 060 | |
| Y1007A | 00005 | 00077 | 00000 | 0145 | 074 | |
| TEKOC | 00005 | 00003 | 00000 | 0147 | 036 | |
| MPSYS | 00007 | 00024 | 00000 | 0147 | 042 | |
| MPSYS | 00007 | 00024 | 00000 | 0147 | 090 | +001 |
| MPSYS | 00007 | 00024 | 00000 | 0148 | 042 | +002 |
| MPSYS | 00007 | 00024 | 00000 | 0148 | 090 | +003 |
| MPSYS | 00007 | 00024 | 00000 | 0149 | 042 | +004 |
| MPSYS | 00007 | 00024 | 00000 | 0149 | 090 | +005 |
| MPSNST | 00003 | 00024 | 00000 | 0150 | 042 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| MPSNST | 00003 | 00024 | 00000 | 0150 | 090 | +001 |
| ASKQ | 00004 | 00039 | 00000 | 0151 | 042 | |
| MP007C | 00004 | 00131 | 00000 | 0152 | 024 | |
| RTST | 00004 | 00150 | 00000 | 0154 | 094 | |
| Y7897S | 00004 | 00007 | 00000 | 0158 | 010 | |
| Y7891S | 00004 | 00004 | 00000 | 0158 | 024 | |
| Y7898S | 00004 | 00003 | 00000 | 0158 | 032 | |
| Y78910 | 00005 | 00001 | 00000 | 0158 | 038 | |
| Y78980 | 00005 | 00002 | 00000 | 0158 | 040 | |
| OLCM1 | 00004 | 00004 | 00000 | 0158 | 044 | |
| OLTR1 | 00004 | 00001 | 00000 | 0158 | 052 | |
| Y78970 | 00005 | 00004 | 00000 | 0158 | 054 | |
| ASKQ1 | 00004 | 00037 | 00000 | 0158 | 062 | |
| OLTR | 00004 | 00002 | 00000 | 0159 | 040 | |
| CSORTC | 00004 | 00025 | 00000 | 0159 | 044 | |
| BASYS | 00007 | 00024 | 00000 | 0159 | 094 | |
| BASYS | 00007 | 00024 | 00000 | 0160 | 046 | +001 |
| BASYS | 00007 | 00024 | 00000 | 0160 | 094 | +002 |
| BASYS | 00007 | 00024 | 00000 | 0161 | 046 | +003 |
| BASYS | 00007 | 00024 | 00000 | 0161 | 094 | +004 |
| BASYS | 00007 | 00024 | 00000 | 0162 | 046 | +005 |
| BASNST | 00003 | 00024 | 00000 | 0162 | 094 | |
| BASNST | 00003 | 00024 | 00000 | 0163 | 046 | +001 |
| SOGJOB | 00007 | 00024 | 00000 | 0163 | 094 | |
| $ASKQ1 | 00005 | 00016 | 00000 | 0164 | 046 | |
| OLTR2 | 00004 | 00002 | 00000 | 0164 | 078 | |
| OLCM7 | 00004 | 00004 | 00000 | 0164 | 082 | |
| OLTR7 | 00004 | 00001 | 00000 | 0164 | 090 | |
| OL007A | 00004 | 00055 | 00000 | 0164 | 092 | |
| OLTR0 | 00004 | 00002 | 00000 | 0166 | 010 | |
| OL007B | 00004 | 00056 | 00000 | 0166 | 014 | |
| MP007B | 00004 | 00130 | 00000 | 0167 | 030 | |
| OL002A | 00004 | 00092 | 00000 | 0170 | 002 | |
| $Y0003 | 00005 | 00003 | 00000 | 0171 | 090 | |
| $ASKQ | 00005 | 00015 | 00000 | 0172 | 000 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| #Y0001 | 00005 | 00010 | 00000 | 0172 | 030 | |
| Y10070 | 00005 | 00080 | 00000 | 0172 | 050 | |
| #MPINP | 00005 | 00009 | 00000 | 0174 | 018 | |
| MPINP | 00004 | 00029 | 00000 | 0174 | 036 | |
| Y0001 | 00004 | 00069 | 00000 | 0174 | 094 | |
| #DCP | 00005 | 00010 | 00000 | 0176 | 040 | |
| #DCP | 00005 | 00010 | 00000 | 0176 | 060 | +001 |
| #DCP | 00005 | 00010 | 00000 | 0176 | 080 | +002 |
| PCSJOB | 00007 | 00100 | 00000 | 0177 | 004 | |
| DCPSYS | 00007 | 00024 | 00000 | 0179 | 012 | |
| DCPSYS | 00007 | 00024 | 00000 | 0179 | 060 | +001 |
| DCPSYS | 00007 | 00024 | 00000 | 0180 | 012 | +002 |
| MP002B | 00004 | 00165 | 00000 | 0180 | 060 | |
| Y10020 | 00005 | 00060 | 00000 | 0184 | 006 | |
| RAF | 00007 | 00024 | 00000 | 0185 | 030 | |
| RAF | 00007 | 00024 | 00000 | 0185 | 078 | +001 |
| RAF | 00007 | 00024 | 00000 | 0186 | 030 | +002 |
| RAF | 00007 | 00024 | 00000 | 0186 | 078 | +003 |
| MP002A | 00004 | 00165 | 00000 | 0187 | 030 | |
| MP002C | 00004 | 00165 | 00000 | 0190 | 072 | |

: